Audience Response Clicker Project

Joseph Meyer
Western Michigan University, joseph.w.meyer@wmich.edu

Follow this and additional works at: http://scholarworks.wmich.edu/honors_theses
Part of the Other Computer Sciences Commons

Recommended Citation
This report will address the feasibility of a software-based audience response system to be implemented, to be known as the Clicker Project. The project seeks to address the growing need for active learning environments at the University level. The primary goal of the project is to construct a web-based system that allows presenters to gather responses from a viewing audience. To this end, presenters can collect and store data dynamically from whomever is in attendance. The following sections will discuss the project in greater detail.
# Table of Contents

1. Abstract ....................................................................................................................... 1
2. Introduction .................................................................................................................. 4
3. Background Section .................................................................................................... 5
   4.1 Client ....................................................................................................................... 5
   4.2 Environment .......................................................................................................... 5
   4.3 Existing Solutions ................................................................................................. 12
      4.3.1 Demonstration of PolLEV’s SMS capabilities ................................................. Error! Bookmark not defined.
   4.4 Problems ................................................................................................................. 13
4. Design Decisions ........................................................................................................ 15
   5.1 Framework ............................................................................................................ 15
   5.2 Object Relational Mapper ...................................................................................... 15
   5.3 PHP Programming Standard .................................................................................. 16
   5.4 PHP Unit Testing .................................................................................................... 16
5. Design .......................................................................................................................... 17
   6.1 Zend Framework .................................................................................................... 17
   6.2 Relational Database ............................................................................................... Error! Bookmark not defined.
   6.3 User Roles ............................................................................................................. 19
   6.4 Authentication ........................................................................................................ 20
   7.1 Functional .............................................................................................................. 25
   7.2 Requirements ......................................................................................................... 26
   7.3 Later Releases ......................................................................................................... 28
7. Resources ..................................................................................................................... 30
8. Feasibility ..................................................................................................................... 30
10. Legal ........................................................................................................................... 32
11. References .................................................................................................................. 33
3 INTRODUCTION

This report will address the feasibility of a software-based audience response system to be implemented, to be known as the Clicker Project. The project seeks to address the growing need for active learning environments at the University level. The primary goal of the project is to construct a web-based system that allows presenters to gather responses from a viewing audience. To this end, presenters can collect and store data dynamically form whomever is in attendance. The following sections will discuss the project in greater detail.
4 BACKGROUND SECTION

This section will address the history of this project to provide context for its need. The client, existing solutions, and problems with existing solutions will be addressed in order.

4.1 CLIENT

Dr. John Kapenga is a faculty member in the Department of Computer Science at Western Michigan University. His idea for the Clicker Project stemmed from his desire to engage students in the classroom more, through pop quizzes, interactive lectures, and roll call. However, the use of the system would not be limited to a single faculty member. Dr. Kapenga hopes that such a system would be used by the entire Computer Science Department in a variety of CS courses, especially larger, entry-level classes required by the CS curriculum. Upon delivery of the product, and initial deployment within the CS department at WMU, the client will assess the possibility of extending the program to be used campus-wide in additional learning curriculums.

4.2 ENVIRONMENT

The Department of Computer Science at Western Michigan University was established in 1979 as an off-shoot branch from the Mathematics Department. The department’s primary mission is to advance the field of computer science through teaching, research and service. In recognition of the changing learning styles within the student population, the Computer Science Department (along with other departments on campus) have acknowledged the increasing need to keep students dynamically engaged in classroom lectures. The university has since invested in the i-clicker system, which is a tool for integrating one type of active learning in the classroom –
audience polls. Active learning is a term that refers to a broad spectrum of educational techniques that focus on engaging students in the learning process [1]. Tangent to traditional lectures and homework, active learning techniques often focus on having students participate in the teaching process through activities such as group discussion and debate, learning groups, and lecture interaction. According to proponents of the active learning, students are more likely to retain knowledge if they are more involved in the learning process, especially with they apply their existing knowledge and make logical connections between topics discussed in class. Furthermore, instructors gain much more immediate feedback on whether or not their material is being understood, so they can adjust their teaching plan on the fly. The Computer Science Department has already placed a strong focus of paired programming work in introductory courses at WMU, and many courses at both the College of Engineering and Applied Science, and the rest of WMU have shifted the focus from a standard lecture format to a more discussion-based classroom environment, featuring smaller class sizes and more instructor interaction.

The i>clicker system (and other similar systems) serves to engage students in classroom. The system consists of a remote (one for each participating student), a hub receiver, and a laptop (typically provided by the instructor) and some type of display. Typically, an instructor will construct a series of questions in advance to show to the students. As is the case with i>clicker, questions are frequently constructed outside the system in another application (such as Microsoft Powerpoint). When the instructor shows a question and tells the i>clicker system to begin accepting responses via a desktop interface, the hub counts the most recent button pressed by each student remote in the room.
After each question, the instructor may display the results of the poll in the form of a bar graph. If remotes are registered to specific students, the instructor may review the scores of students and export them to a file for uploading to another system, including web systems implemented at WMU.

Currently, Western Michigan University implements an online course system called Desire to Learn (or D2L). This web interface provides a hub in which students may take online quizzes and exams, view content posted by the course instructor and convene electronically with their peers to discuss course topics. The quiz section of this system is particularly interesting and will be used as a general model for structuring the Clicker Project. By mirroring the current design
layout, faculty will have less content to relearn when the new system is deployed in the future.

Figure 4-2: D2L Quiz Organization

Faculty especially appreciate the ability to organize questions to their liking, as pictured below:

Figure 4-3: D2L Demonstration of customizable folder hierarchy
Emulating this feature was a high priority for this project. The current system does however lack some key, useful features. First of all, the system does not allow for polling audiences in a “live” sense – each participant takes quizzes and tests within the system at his or her own pace. In some cases, it is more desirable to have an entire audience step through a poll one question at a time. In this setting, the presenter can display the results of each question to the audience, explain the answer, and field questions if needed as they arise. Poll Everywhere – another audience response system – provides less functionality than D2L, but does offer some of these key features. Furthermore, a more flexible system can be used to quickly take attendance and gather feedback. D2L lacks mobility: the creation of quizzes and polls can be quite involved and time consuming. By streamlining the quiz/poll creation process, instructors will be more able to rapidly assimilate the system within their environment and utilize the system accordingly.

The environment within the Computer Science Department is very conducive for a project like this. The department owns and maintains a set of servers which typically provide development environments to students who connect via SSH, as well as host department resources such as websites and repositories. Additionally, as department policy, all students entering the CS program at WMU are required to have a laptop [2]. While the end goal of the project is to implement the system in a much broader sense, field testing it first with the computer science department can allow for less risky, but valuable assessment of the stresses the system will need to bear.

In addition to laptops, many students have access to smartphones, tablets and other devices that can access the web. This creates an excellent environment in which students may participate in class on hardware that they already own, eliminating the need for task-specific
hardware. In an effort to accurately evaluate web-enabled device ownership among the student population, a survey was distributed to roughly 100 students attending class at WMU. Participants were enrolled in a variety of colleges on campus, including (but not limited to) Health and Human Services, Fine Arts, Business, and Engineering. The following chart demonstrates the breakdown of enrollment of the respondents:

![Figure 4-4: Distribution of colleges survey respondents are enrolled in](image)

Of the student’s surveyed, the vast majority reported they owned at least 1 web enabled device, the most popular being a laptop. Many students had, in addition to a laptop, a web-capable cell phone or tablet. The survey also asked students to list which devices they commonly used in or brought with them to class. The following chart details the results of the survey regarding smart device ownership:
It should be noted that the pool of students surveyed typically owned a laptop AND a smart phone, but brought their phone with them to class more often than their laptop. Additionally, of the students surveyed only 2% responded saying that they did not own any web-capable device that could be considered mobile (such as a desktop computer), meaning that the odds of students owning some kind of device already that could access the web are very good. Of the web-enabled devices students at WMU used, the survey showed the following results regarding what web browser students used to access the web:

![Bar chart showing web browser preferences](image)

*Figure 4-5: Student web-enabled device ownership statistics*

*Figure 4-6: Survey demonstrating student web-browser preference*
Lastly, the survey asked students what operating system best described the device they most commonly use to access the web.

![Survey showing student OS preference]

**Figure 4-7: Survey demonstrating student OS preference**

Though the end goal of the project is widespread usability across all devices, while designing the system the following facts were taken into consideration:

1. Students will most likely be using the system on a device screen smaller than 7 inches diagonal
2. Students will most like be using Chrome as a web browser on a laptop
3. Students will most likely be using Apple Webkit on mobile phones

### 4.3 Existing Solutions

A handful of solutions already exist for this type of classroom solution. One system adopted at WMU is the hardware-based i>clicker system (mentioned previously), which relies on student remotes and a hub receiver to collect data from a class. [3] Hardware-based options are quite common, and range in size and function from company to company, however, the end product is...
largely the same. i>clicker also offers software-based implementations for their product, though audience response in digital is rarely offered by competitors. Additionally, Poll Everywhere (or PollEv) has deployed a web-based audience response system, that offers the unique feature of accepting responses via SMS and browser interfaces, meaning audiences can submit responses with phone-based text messages or enter them via a laptop, tablet or phone web browser [4]. PollEv is advantageous for small classes, since entry-level options are free.

![Figure 4-8: Demonstration of PollEV’s SMS capabilities](image)

### 4.4 Problems

A hardware-based system is not desirable because it relies totally on instructors and students acquiring additional, expensive hardware, which can be broken, lost, stolen, etc. The total cost of a hardware-based solution is generally fairly high, though i>clicker is relatively affordable when compared to other options [3]. However, though hubs are typically provided for instructors by
the department, students must purchase remotes individually, which may need to be replaced if
the university upgrades their system. Furthermore, the remotes are discreet enough that
students can bring in multiple remotes easily and submit answers for their peers. That being
said, much of the system is already software based, and the makers of i>clicker have produced a
hardware independent option to their existing solution. However, this system requires
additional license fees and lacks features the client desires. Other software-based solutions exist
but are very feature limited and face the same expense problems as the i>clicker system. PollEv
is one such solution, but restricts class sizes to 40 students and limits reporting capabilities
unless a premium account is purchased.

Shifting the system to a web-based hardware independent platform can reduce many of the
costs associated with implementing a clicker system by simply allowing instructors and students
alike to use devices that they already own for work or for personal use. Additionally, by shifting
the development of the project to students, a new system lacks the licensing fees associated
with a professionally developed system. Additional costs considerations need to be taken into
account regarding powering, operating, and maintaining the server where the application will
reside, but relative to the cost of purchasing (or replacing) and distributing remotes and hubs
necessary to the current model, the benefit of a web-based solution is still more financially
viable.
5 DESIGN DECISIONS

This section will address the selection process used to choose key components of the project.

5.1 FRAMEWORK

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Visibility</td>
<td>Open</td>
<td>BlackBox</td>
<td>Open</td>
<td>Open</td>
<td>Open</td>
</tr>
<tr>
<td>Documentation</td>
<td>Very High</td>
<td>High</td>
<td>Low</td>
<td>Medium</td>
<td>Medium</td>
</tr>
<tr>
<td>Developer</td>
<td>Zend Tech.</td>
<td>Cake Software</td>
<td>Communit</td>
<td>Ellis Lab, Inc</td>
<td>Sensio Labs</td>
</tr>
<tr>
<td>Learning Curve</td>
<td>Medium</td>
<td>Medium</td>
<td>Low</td>
<td>Low</td>
<td>Medium</td>
</tr>
<tr>
<td>Popularity</td>
<td>Very High</td>
<td></td>
<td></td>
<td>Medium</td>
<td></td>
</tr>
<tr>
<td>License</td>
<td>New BSD</td>
<td>MIT</td>
<td>MIT</td>
<td>OSL v 3.0</td>
<td>MIT</td>
</tr>
</tbody>
</table>

ZF2 – Zend Framework 2 – was chosen as the framework. Other key deciding factors included the quality and availability of documentation, and popularity among web developers.

5.2 OBJECT RELATIONAL MAPPER

<table>
<thead>
<tr>
<th></th>
<th>Doctrine 2</th>
<th>Propel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Framework</td>
<td>ZF2, et al</td>
<td>Many</td>
</tr>
<tr>
<td>Popularity</td>
<td>Very High</td>
<td>Medium</td>
</tr>
<tr>
<td>Documentation</td>
<td>Very High</td>
<td>High</td>
</tr>
<tr>
<td>Implementation</td>
<td>Data Mapper</td>
<td>Active Record</td>
</tr>
</tbody>
</table>
Doctrine 2 was selected as the Object Document Mapper, largely due to the availability documentation directly related to ZF2.

### 5.3 PHP Programming Standard

*Pear [10]*

<table>
<thead>
<tr>
<th>Framework</th>
<th>ZF2</th>
</tr>
</thead>
</table>

After selecting ZF2 for the project’s framework, Pear became a necessary choice as a programming standard, since ZF2 relies on Pear.

### 5.4 PHP Unit Testing


<table>
<thead>
<tr>
<th>Learning Curve</th>
<th>Medium</th>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Command Line</td>
<td>Web Browser</td>
</tr>
<tr>
<td>Popularity</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Documentation</td>
<td>Eclipse, ZF2</td>
<td>None</td>
</tr>
<tr>
<td>Actively Maintained?</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

PHP Unit was chosen as the Unit Testing framework. Key deciding factors included the availability of documentation and compatibility with the chosen web development framework, ZF2.
6  DESIGN

This section will cover the specific details of each component implemented in the project in
detail. Furthermore, this section will expand upon the advantages of each chosen design facet
and note the difficulties each facet addressed.

6.1  ZEND FRAMEWORK

Zend Framework 2 (ZF2) was selected as the framework. A framework is extremely desirable for
a project like this one. In general, pre-built frameworks provides a uniform structure on which to
develop. A framework enforces coding standards by forcing developers to adhere to a set file
hierarchy. Furthermore, most frameworks provide built-in libraries that assist in tasks like form
validation, input/output filtering, session and cookie handling, etc which greatly expedites the
development of a project. Since many operations are included within the framework, the end
result is a more stable program that doesn’t rely on third-party plug-ins. In the long run,
frameworks tend to reduce the amount of code needed for a project and makes an application
easier to maintain in the long-run; all files included in the project have obvious locations within
the hierarchy, which lends itself nicely to functional and unit testing. Zend relies on what is
known as a Model View Controller structure (MVC). [13] An MVC is a type of architecture that
separates the representation of information from the user’s interaction with it.
The controller serves to mediate input, which it converts to commands that can be interpreted by the model. These commands are then relayed to a server which has access to a database backend. The response is then passed back to the model and again to the controller which renders a view for the client to see. [14] For the implementation of this project, the “model” component has been replaced by the Doctrine 2 equivalent, an “Entity”. However, the entity serves much the same purpose: mediating application requests sent to the server and managing responses sent from the server to the application.
6.2 User Roles

For additional context, it is necessary to outline a flow of user actions for each access level of the application (Student, Instructor, and Administrator). The global relationship between each level of access roles associated with each level of user is as follows:

![User Access Levels and Capabilities Diagram]

*Figure 6-2: User access levels and capabilities*

It is important here to distinguish the capabilities of each different type of user. Administrators act only to promote (and demote) standard level users (called Students) to Instructors. When a user is promoted to Instructor, they gain access to quiz builders, course rosters, and other
features critical to administering a clicker quiz to a class. Instructors may view class results of an administered quiz and export the results to excel. While Instructors create courses and build content within the course, it is up to the Student to register for the course using a key provided by the application (displayed to the instructor on their home page).

By far, the flow of actions made by the Student are the most simple. Once a student’s login credentials are validated, the student will be able to access their list of courses and access whichever courses they have registered for via a drop down menu. Once a course has been selected, the student will wait at the response screen for a question/quiz to be administered before submitting a response. Once a question is opened to response, students may then select an answer to submit for grading. Global responses, correct answers, and answer explanations may be presented to the student at the instructor’s discretion via projector screens provided in most university classrooms. If a student chooses not to select a course, they may instead review previous quiz results via a link from the homepage.

Guests have the same features as students, but may not review their previous responses and their data is not stored in a gradebook. Guest responses will, however, appear in an Instructor’s histogram when reviewing responses from an administered question.

6.3 Authentication

It is important to note that the user must first pass through a validation process to access the application. In this implementation, the application relies on LDAP authentication, which allows students to use the same username and password they typically use to access WMU resources to gain admittance to this system. For non-standard users (particularly administrators), the
password is determined on an individual basis. Administrator passwords are protected via
BCrypt Password Hashing. LDAP passwords are not stored in the application database. Guests,
any user not in the LDAP system who would like to participate, need not provide credentials to
access the application. Instead, the Guest relies totally on generated session variables to
navigate through the application.

6.4 INTERFACE

User Interface design was important during the design process. The application seeks to
streamline many of the superfluous options available in D2L and overall, reduce the amount of
“stuff” the user would need to sort through to use the application. Furthermore, by using the
same design as much as possible for both student and instructor, the project ensures that a user
can operate smoothly when transitioning from one role to another.

![Figure 6-3: A directory of courses is presented similarly to both students and instructors]
When building quizzes for a class, instructors are given the opportunity to design their own folder organization in which to arrange their material. In this way, the application does not impeded organizational methods an instructor has already established and is much more comfortable to use. What content is displayed in the body of the page is dependent on what options are selected in the sidebar navigational pane.

Figure 6-4: Left - Sidebar navigation, Middle - Quiz editor, Right - Question editor
When administering a quiz to a class, the following screen is presented to the instructor, likely for display on in-room projectors.

![Quiz in progress](image)

*Figure 6-5: Quiz in progress*

Students are presented with a screen that serves the same function as the remotes in the i>clicker model.
Upon completing the quiz, the instructor may display the correct answer of their page, or return to a quiz menu page to review the results graphically. Results may also be observed from a gradebook page, which distinguishes between students who answered correctly/incorrectly vs students who did not respond. This feature can be useful for taking attendance.

Likewise, students may review their individual results from a gradebook page.

*Figure 6-6: Chart displaying response results*
7 STORIES

Discussions between the client and the programming team are held to determine what features of a project are most critical. These interactions are distilled into “stories” – each of which describes a very specific aspect of what is needed to make the project complete. This section will review the stories selected as most important for Release 1.

7.1 FUNCTIONAL

Functional stories are those which detail the aspects of how the final project should behave and function. This section will outline which functional stories were selected for Release 1.

<table>
<thead>
<tr>
<th>STORY</th>
<th>RISK</th>
<th>TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instructors will be able to select questions from a question bank to construct quizzes. Questions can be created in advance or on-the-fly when building a new quiz</td>
<td>3</td>
<td>8 hours</td>
</tr>
<tr>
<td>The system should support multiple instructors</td>
<td>4</td>
<td>6 hours</td>
</tr>
<tr>
<td>Instructors will be able to define their own folder organization for the class.</td>
<td>7</td>
<td>20 hours</td>
</tr>
<tr>
<td>Instructors will NOT be able to</td>
<td></td>
<td></td>
</tr>
<tr>
<td>While administering a question, the instructor should have the option of viewing a histogram of responses that populates dynamically as students reply, which could then be displayed on student screens per the instructor’s discretion.</td>
<td>4</td>
<td>4 hours</td>
</tr>
<tr>
<td>The instructor should also be able to view the correct answer to the question,</td>
<td>3</td>
<td>2 hours</td>
</tr>
</tbody>
</table>
countdown timer (if the question is timed) and the running total of students
responded vs. students registered.

Questions will be of multiple choice format, in ASCII text

<table>
<thead>
<tr>
<th>STORY</th>
<th>RISK</th>
<th>TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>After a question is administered, instructors will be able to review the results in a gradebook. The results can be exported to excel and the resulting spreadsheet should be compatible with D2L, allowing Instructors to import their grades into the system.</td>
<td>3</td>
<td>8 hours</td>
</tr>
<tr>
<td>The system should be able to support attendance taking methods: that is, the system will differentiate between students who responded incorrectly, versus students who responded correctly, AND students who did not respond at all</td>
<td>1</td>
<td>2 hours</td>
</tr>
</tbody>
</table>

### 7.2 Requirements

Requirement stories address the needs of a project, but no particular function. This section will outline the Requirements for Release 1.

<table>
<thead>
<tr>
<th>STORY</th>
<th>RISK</th>
<th>TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>The system should be able to support 1 class of 30 students for the core release. To support, larger lectures, the system should be able to process the responses of up to 200 students from any 1 class and be capable of supporting multiple classes simultaneously.</td>
<td>4</td>
<td>15 hours</td>
</tr>
<tr>
<td>Project will adhere to coding standards that are defined to maximize compatibility across the most recent versions of common web browsers: IE,</td>
<td>4</td>
<td>8 hours</td>
</tr>
</tbody>
</table>
Firefox, Safari, and Chrome. Coding standards should keep in mind variable screen sizes and resolutions so that the project will render appropriately on mobile devices as well as laptops.

Project will be designed to run on an Apache server using php and MySQL. The project – if not initially deployed on a department-based server – should be easily deployed to a server in the CS department. See the section of Resources below for information of software versions.

Students should be able to easily respond on all devices, including both mobile and traditional web browsers. That is, the experience of a Student responding should be identical on a mobile device such as a cell phone or tablet and a laptop.

Instructor interface should be confined to laptop and desktop browsers.

We will address the likelihood of students being able to see chosen responses on the screens of nearby devices. Cheating will be addressed with increasing significance each release.

Authentication will be provided in two ways in the system. Standard users will be allowed to authenticate using LDAP credentials as defined by the university. Special users, system Administrators, will login via encrypted passwords created by the user individually. Only encrypted passwords will be stored by the application. LDAP credentials will not be stored.
7.3 **LATER RELEASES**

The following stories have been compiled to outline additional features to be included in the project in later releases. These objectives seek to further expand the capabilities of the application and improve its usefulness. Each story is listed by order of importance:
<table>
<thead>
<tr>
<th>STORY</th>
<th>RISK</th>
</tr>
</thead>
<tbody>
<tr>
<td>The system will support textual input besides plain text: This will allow users to construct HTML or otherwise formatted input to create code blocks, mathematical equations, of other methods of differentiation to display to the class during quiz administration.</td>
<td>4</td>
</tr>
<tr>
<td>The system will support images as both question content and answer content, allowing for greater flexibility of questions.</td>
<td>4</td>
</tr>
<tr>
<td>The system will support varied answer formats, including True/False, matching, or any form of multiple choice will 2 or more possible answers. A reasonable limit will be imposed on how many possible answers may be displayed.</td>
<td>2</td>
</tr>
<tr>
<td>Push and Comet functionality will be extended to student devices, allowing question and answer content to be displayed on any device with acceptable screen resolutions to augment the current, label-only display rendered on all devices.</td>
<td>3</td>
</tr>
</tbody>
</table>
8 RESOURCES

Apache Server

MySQL 5 database

PHP 5.3.0 or greater

2 or more devices (recommended)

Network connection

9 FEASIBILITY

After considerable review and the construction of a core system, it has been determined that the project is feasible. Upon delivering the initial release to the client, the application will be inspected to verify its validity and assess the possibility of further development.
10 GLOSSARY

**Audience/Student Response System** – a system used to collect the responses from audience members when a host or instructor asks a question, administers a group quiz, takes attendance, etc.

**Hardware Independent** – any solution that does not rely on a specific piece of electronic hardware to operate.

**PHP** – PHP: Hypertext Preprocessor is a server-side scripting language. PHP scripts are executed on the server and is free to download and use.

**Staging Area** – a screen which will allow a presenter to navigate directly to desired actions, such as constructing a question or administering a quiz.

**Course** – a class or teaching environment. May also be applied to lectures and presentations

**Zend Framework** – a pre-built website structure and subsystems

**Student** – an individual who will be responding to course polls and quizzes

**Instructor** – an individual who will be administering course polls and quizzes. A presenter. Typically whomever’s speech is the focus of the course

**Administrator** – an individual who manages the entire clicker System. An administrator does not necessarily interact with instructors or students directly.

**MySQL** – Popular open-source database for web applications
11 LEGAL

Per the client, this solution is to be an open source option under the MIT licensing. Western Michigan University will own the software and distribute it as needed. No NDA is needed, as this project does not directly handle sensitive information.
12 References


[2] Department of Computer Science, "Laptop Requirements".


