UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE
BSC COMPUTER SCIENCE WITH BUSINESS AND MANAGEMENT

The Fastbleep Schools Booking System

Third Year Project Report

Author: Oana Bradulet
Project Supervisor: Prof. John Gurd

APRIL 2013

Abstract

Fastbleep is a Manchester-based social enterprise, operating in the healthcare education field. Their Schools initiative is aimed at widening participation of under-represented groups in medical schools. A big part of this division's activity is represented by school sessions – interactive workshops led by university students for pupils.

Due to rapid growth and increasing demand, the organization urgently needs an IT system for the booking and management of school sessions.

This report documents the end-to-end development of a web application designed to facilitate and enhance the activities of Fastbleep Schools. Following an overview of the organization and the project, a background into the concepts and paradigms related to software engineering and web application development is provided.

The project was developed using an iterative Agile approach, with emphasis on strong user engagement, continuous refinement of requirements and design, incremental implementation, and early testing. While outlining each of these activities in some detail, the report also highlights key aspects and critically analyses them.

The conclusion represents a reflection over the project overall. Since the author's engagement with the Fastbleep organization does not end here, next steps for the application are briefly described.

• • •

Project Title: *The Fastbleep Schools Booking System*

Author: *Oana Bradulet*

Project Supervisor: *Prof. John Gurd*

Word Count: *10,721*

• • •

Acknowledgements

Firstly, I would like to thank my project supervisor, Prof. John Gurd, for his continued support and valuable guidance throughout this project.

I would also like to thank the Fastbleep team, for the opportunity to take part in their remarkable work. Their approachable and encouraging attitude has made the collaboration very pleasant and ultimately successful.

Last, but not least, I must thank my peers – Radu Asandei and Qasim Ashraf – and their project supervisor – Prof. Andrew Brass – for their input and co-operation.

Table of Contents

List of Figures

List of Tables

1 Introduction

This chapter provides a high-level overview of Fastbleep and the project. Following an outline of the background and context within which the organization operates (Sections 1.1 and 1.2), the problem (Section 1.3) and the objectives of the proposed solution (Section 1.4) are described. The chapter concludes with a brief summary of the remaining contents of the report (Section 1.5).

1.1 Widening Participation in Medical Schools

Securing equal opportunities and fair access to Higher Education (HE) for all is a key strategic goal for the British Government [1]. Currently, however, there remain a number of barriers to access, particularly in the case of certain socio-economic groups. For example, students from disadvantaged backgrounds and those with disabilities and learning difficulties are significantly under-represented in HE Institutions. Therefore, widening participation is a focus not only from the perspective of the UK's Department for Employment and Learning, but also for universities looking to increase diversity among their students and ensure admittance of the best candidates, regardless of their personal or social backgrounds.

In the case of medical schools in particular, research has shown that students from deprived areas and lower socioeconomic circumstances are severely under-represented, not only in applications but also in acceptance rates [2] [3]. The barriers come not simply in the form of financial hardship or academic inability, but mostly due to pupils' "stereotyped and superficial" perceptions. They see medical schools as "distant, unreal, and culturally alien" [4] and cannot identify themselves with the "elite image" of medicine [5]. Furthermore, those pupils that did show prospective interest, lacked access to suitable support and resources needed for a successful application [2].

In consequence, initiatives and strategies aimed at encouraging pupils from non-traditional groups into HE and medical schools are considered necessary and a political priority in the UK [6]. By seeking to alter the general misconceptions regarding medical education and the profession, these schemes have a potentially high and very beneficial impact on local communities as well as HE institutions and society at large.

1.2 Fastbleep and Fastbleep Schools

Fastbleep is a non-profit social enterprise in the healthcare and life-sciences education field. Founded by five Medics and one Computer Science student in Manchester in 2010, the organization has a clear, self-declared aim: "to promote learning, creativity and discovery for all" [7].

Fastbleep started out with the Medical Notes tool [7], a collection of articles written by and for medical students. Over the last three years, however, the organization has expanded rapidly to include a multitude of other projects. The member base has grown correspondingly, having reached thousands of followers and contributors both in the UK and abroad.

Fastbleep Schools (FS) is the division directed at widening participation within medical education, based in Manchester [7]. Their main focus is currently represented by School Sessions – interactive events led by medical students and targeted at 11-18 year old pupils. Figures 1.1 and 1.2 illustrate typical activities within such sessions – hands-on experiments meant to spark an interest in pupils about medicine. Additionally, the sessions include informal discussions about applying to medical school and developing a career in the field.

Figure 1.1: Fastbleep Schools Volunteer Teaches Suturing

Figure 1.2: Analysing the Human Skull

The Schools program has grown rapidly over a relatively short period of time. There is increasing demand from schools and colleges around Greater Manchester [7]. The volunteer base has exceeded 150 members. Plans for the near future include geographical expansion (Liverpool, Leeds) as well as spreading out into other disciplines (Biology, Dentistry).

1.3 Project Proposal

It is a recognized fact that any organization requires IT systems and tools not only to carry out day-to-day operations efficiently, but also to enhance and improve its activities. In the case of Fastbleep and Fastbleep Schools, the need for such tools was particularly stringent, given their rapid expansion. As part of the on-going collaboration between the School of Medicine and that of Computer Science, Fastbleep have requested assistance for their growing IT needs.

In accordance with standard practice, three students have been assigned to work with Fastbleep Schools as part of the third year project allocation. Initially, the specific requirements were not clearly defined. Therefore, a thorough analysis has been carried out to identify and prioritize the areas that required immediate attention. Several workshops with the core Schools team and discussions with the project supervisors have resulted into a division of the work that met the main needs of the client as well as the individual skill sets and aims of the students. Table 1.1 illustrates this distribution.

Table 1.1: The Division of Work

Student	Problem	Solution
Qasim Ashraf	In order to obtain further funding, Fastbleep need the ability to evaluate and demonstrate the impact of their interventions.	An impact assessment tool for storage and reporting (Q-Sort).
Radu Asandei*	The current website, that teachers and pupils access, is highly fragmented, confusing, and off-putting.	A consistent, interactive, user-friendly graphical interface.
Oana Bradulet (myself)	The administration of school sessions is done manually, inefficiently and incurs significant overhead in the form of resources and time.	A simple, effective tool that allows the booking and administering of school sessions.

*the deliverable for this project has changed subsequently.

1.3.1 The Fastbleep Schools Booking System Project

Initially, there existed no unified system or tool for the management of school sessions. With the growth in demand and in volunteers, the process of organizing sessions had become a cumbersome task, requiring significant time and effort. The organization did not have capacity for growth or improvement due to resources being utilized in administrative rather than value-added activities.

In addition to inefficiency, the organization lacked capabilities that were essential for sustaining and expanding its activities. For example, there was no consistent tracking of data regarding the organization's operations over time. Additionally, feedback was not formally captured so there was little input in the format of the sessions from teachers or pupils.

In order to resolve these issues, an integrated, database-driven, user-based system was designed and built. The functionality would be added to the current Fastbleep Schools website rather than a completely new system being built. This was the client's specific request for reasons of continuity and ease of adoption.

Given the existence of the second project (Radu Asandei's), the focus is on the raw functionality of the solution. A highly functional user interface remains, therefore, out of scope.

1.4 Project Objectives

The following objectives have been set for the project, in agreement with Fastbleep Schools:

I. Ensure that user requirements and expectations are met to the highest possible extent

Ensure user engagement and continuous refinement of requirements throughout the project lifecycle. The client's overall satisfaction and ultimate practical usability of the tool within the organization are considered key success metrics.

II. Increase efficiency of the management of schools sessions

The application should contribute to the efficiency of activities, and not add unnecessary overhead to the current manner of operation.

III. Build something that is easy to use and maintain

The application should not be overly complicated as it aims to reduce the time and effort needed to carry out tasks. It should not, for example, require user training. Additionally, there is currently no permanent IT support in the organization and fairly little technical expertise. Therefore, the application should be easy to maintain on a day-to-day basis.

IV. Build something that can be integrated within current systems

As mentioned in Section 1.3.1, it was the client's request for a tool that can act as an extension to the functionality of the current website.

1.5 Report Structure

The remaining contents of the report are structured as follows:

Chapter 2: Background

Provides a theoretical background to the concepts and notions utilized over the project lifecycle and referred to in the report.

Chapter 3: Requirements Gathering and Analysis

Presents the choice of development methodology and the reasoning behind it before proceeding to describe the project stakeholders. An analysis of the current ("As-Is") system is undertaken prior to defining the requirements of the new ("To-Be") system.

Chapter 4: Design

Describes the design and architecture of the proposed solution, as well as providing justification for the decisions made. A key step in the design process is represented by evaluating the current infrastructure.

Chapter 5: Implementation

Explains the choices made in terms of tools and technologies and covers the implementation of the system, highlighting the key aspects.

Chapter 6: System Walkthrough

Takes the reader through the major features of the system, showing how they are intended to work in practice.

Chapter 7: Testing and Evaluation

Provides a description of the testing practices undertaken and presents the respective results. Includes a critical analysis of the project achievements.

Chapter 8: Conclusions and Next Steps

Concludes the report, by highlighting the major challenges faced, the lessons learned and by providing a high-level summary. The final section describes the next steps for the application.

2 Background

This chapter provides a background for the various concepts and technologies utilized in the project and referred to throughout this report. The different sections cover: the development methodologies considered for the project (Section 2.1); general web application architecture theory (Section 2.2); design and architecture patterns used in the construction of the system (Sections 2.3 and 2.4); and finally a review of the technologies utilized or considered for it (Sections 2.5 and 2.6).

2.1 Development Methodologies

A software development process or methodology describes "an approach to building, deploying, and possibly maintaining software" [8].

Two key methodologies were considered. FOLDOC (the Free On-Line Dictionary of Computing) defines **the sequential lifecycle (Waterfall)** as a software lifecycle model, in which development proceeds linearly through the conventional phases: requirements, design, implementation, testing, integration, and maintenance [9].

In contrast, **the iterative lifecycle (Agile)** is based on "the successive enlargement and refinement of a system through multiple iterations, with cyclic feedback and adaption as core drivers to converge upon a suitable system" [8].

The choice between the two methodologies is explained in Section 3.1, based on an analysis of their suitability for this project.

2.2 Web Application Architecture

A **web application** is defined as "a client / server application that uses a Web browser as its client program, and performs an interactive service by connecting with servers over the Internet" [10]. As opposed to a traditional web page or site, which simply renders static content, a web application displays dynamically tailored content.

Figure 2.1: Web Application Architecture (taken from **[11]**)

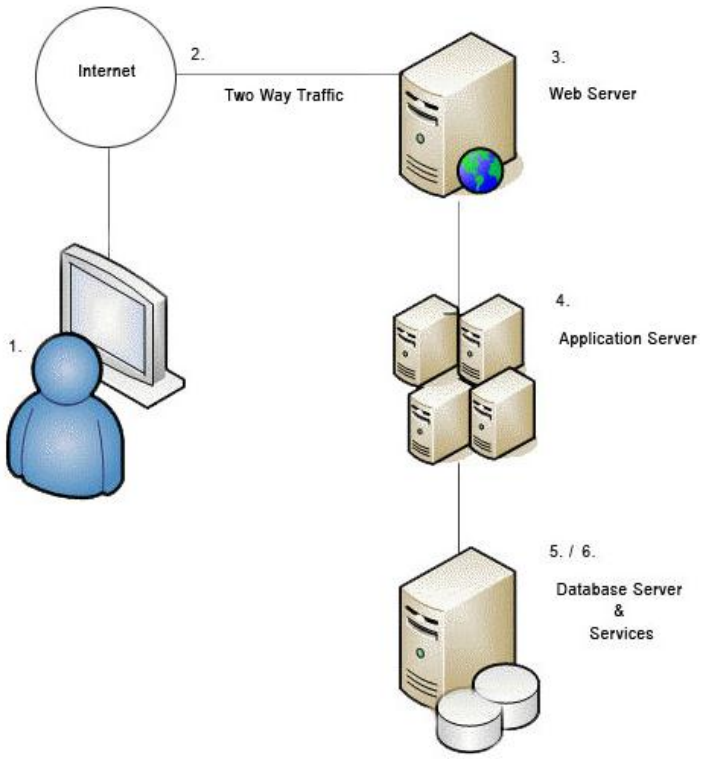
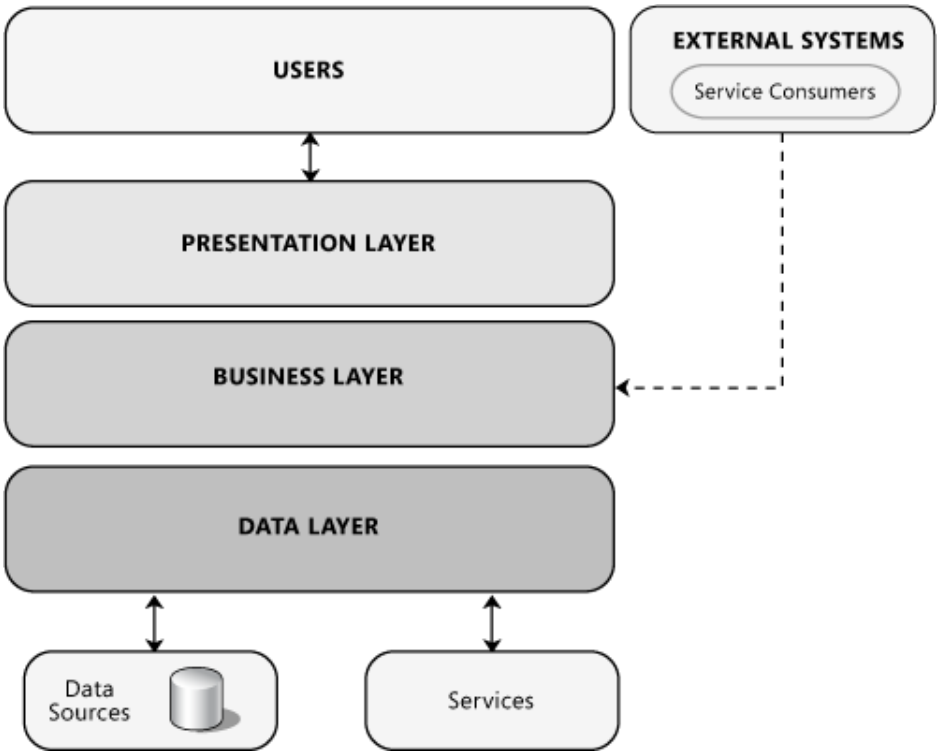


Figure 2.1 depicts a simplified version of how web applications typically work [11]. A user makes a request via a browser. The request is forwarded by the web server to the appropriate application servers which perform the requested task with the help of database servers and services. The response is then delivered to the user via the web server.

The architecture of most web applications is described as “multi-tiered”. As opposed to the traditional client-server architectures, web applications consist of multiple layers (“tiers”) [10].

The most widely spread type of multi-tier or “n-tier” architecture is the three-tiered model (Figure 2.2).

Figure 2.2: Three-Tier Architecture (taken from **[12]**)



The **presentation layer** is the top-most tier, which effectively manages the user’s interaction with the system. The **business layer** is where the core functionality of the application is implemented, along with the relevant business logic. The **data layer** enables the system’s access to internal data (databases that are part of the system) and external data (potentially accessed by means of services) [12].

2.3 Design Patterns

Low coupling is a GRASP (General Responsibility Assignment Software Patterns) design principle for software development which states an element (class, subsystem etc.) should not be dependent or rely on many other elements [8]. This way, changes are easier to implement and the elements are easier to reuse.

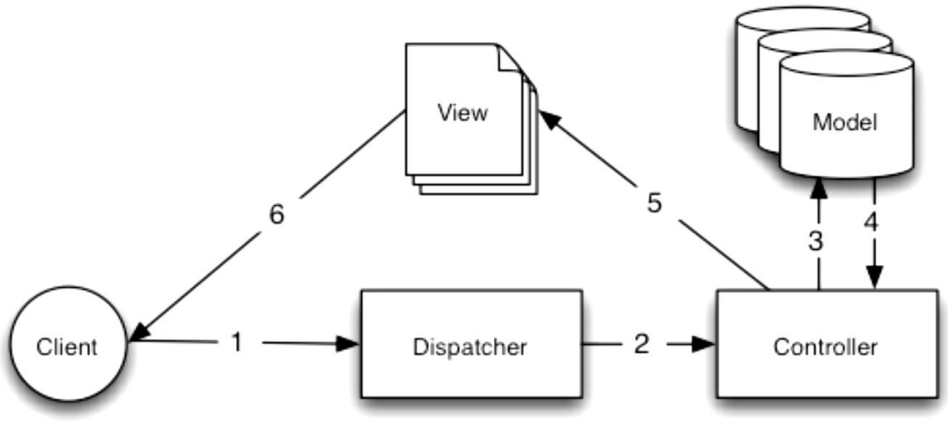
High cohesion is also a GRASP design principle which states that the responsibilities of an element should be highly related and focused and that the element should not do a tremendous amount of work [8]. A cohesive class is then easier to comprehend, reuse, maintain, and less affected by changes in other elements.

2.4 The MVC Pattern

Model-View-Controller (MVC) is an architecture pattern that explicitly separates the user interface from the underlying data representation [13]. The structure of the application is thus divided into three specialized components. The **Model** contains the application data as well as the logic for querying that data as dictated by the user. The **View** is the component that directly interacts with the user and displays the information provided by the model. The **Controller** processes user commands and forwards them to the model or the view, as appropriate [13].

Figure 2.3 illustrates how a client request is handled in a popular development framework that uses the MVC pattern [14].

Figure 2.3: A Typical MVC Request in CakePHP (taken from **[14]**)



The MVC pattern is often (although not always) used in conjunction with a three-tiered architecture – where the Controllers and Views are contained in the Presentation layer and the Models in the Business layer.

2.5 Rapid Development Framework

Rapid development frameworks, in the context of web applications, are foundational structures that aim to assist developers by eliminating the overhead of coding basic, standard web app functionality and allowing them to focus on application-specific logic [14]. Such frameworks typically provide a series of features to facilitate and enhance the application development process. Most of them include, for example, database interaction and the generation of basic CRUD (create, read, update, delete) operations.

Examples of rapid development frameworks include CakePHP, Zend, Kohana, and Codeligniter. The first three have been evaluated in terms of their suitability for the project. They are described in Section 2.6.3; the analysis behind the decision as to which framework to utilize for this project is illustrated in Section 5.1.1.

2.6 Technology Review

The following tools and technologies have been considered or utilized, directly or indirectly, in the construction of the web application. Their specific utilization and the reasoning behind it are more widely discussed in Section 5.1.

2.6.1 Client Side Technologies

HTML (Hyper Text Markup Language) is a language that uses a set of tags and plain text to describe web page content [15].

CSS (Cascading Style Sheets) are style sheets that define how HTML elements are displayed and formatted [16].

Javascript is a scripting language that can be inserted into HTML pages, and executed by any browser [17].

2.6.2 Server Side Technologies

PHP (Hypertext Preprocessor) is a widely-used, open-source scripting language. PHP code is executed on the server and the result is returned to the browser as plain HTML. Typical uses of PHP include generating dynamic content, collecting form data, handling data from the database, restricting user access etc. [18].

MySQL is the most popular database management system used in conjunction with PHP [19]. Free to download and use, MySQL is known to be fast, reliable, and easy to use. An additional benefit is the fact that it can run on multiple platforms. MySQL uses the relational database model: the data is stored in tables, that consist of rows (entities) and columns (attributes); the tables are related to each other via meaningful associations.

The **Apache HTTP Server** is a robust, commercial-grade, freely available web server software program [20]. Developed and maintained by an open community of contributors, Apache is customizable, extensible, and works on a number of different operating systems [21].

2.6.3 Frameworks, Libraries, and IDEs

CakePHP is a free, open-source rapid development framework for PHP [14]. The latest version (2.3) of the tool is compatible with PHP 5.2.8 and greater. It uses the MVC software design pattern and offers developers a wide set of features, such as integrated CRUD for database interaction, code generation, and built-in data validation.

Zend Framework is an open-source framework for developing web applications and services using PHP 5.3 and greater [22]. The framework abides to object-oriented design principles and uses an MVC-based implementation. It is a very popular tool, its benefits including modularity and extensibility.

Kohana is an open-source, object-oriented web application framework built using PHP 5 [23]. It utilizes MVC-based implementation and provides various features, including code generation. The framework is widely known for its efficiency and fast configuration.

JQuery is a Javascript library that aims to simplify common tasks that would normally require extensive Javascript code by wrapping it into simple methods [24]. It is typically used for HTML and CSS manipulation, HTML event handling etc.

NetBeans is an integrated development environment (IDE) that supports a multitude of languages, from Java to C/C++ to PHP [25]. It is free, open-source, and works on various operating systems.

2.7 Summary

This chapter covered a set of key concepts, principles, and tools in software engineering and web application development.

Firstly, two major development methodologies were introduced. The choice of methodology ultimately dictates the practices employed over the entire project lifecycle.

In terms of web applications, a background on their architecture and principles (design and architecture patterns) is essential. The system design (Chapter 4) and implementation (Chapter 5) are shaped by and conform to these norms.

Finally, a set of tools and technologies were briefly described to set the context for their utilization in implementing (Chapter 5) and testing (Chapter 7) the system.

3 Requirements Gathering and Analysis

This chapter starts by describing the development methodology chosen (Section 3.1) and the practices used for requirements gathering and analysis (Section 3.2). Using these techniques, a stakeholder analysis (Section 3.3) and an investigation of the "As-Is" system (Section 3.4) were carried out, leading to a thorough understanding of the "To-Be" system and its requirements (Section 3.5).

3.1 Development Methodology

The Waterfall approach is normally associated with high rates of failure and with systems that ultimately do not meet user requirements [8]. Its suitability for this project was especially questioned due to the existence of "real world" users and the high level of uncertainty. As mentioned in Section 1.3, the customer did not have a clear view from the outset with regards to their IT needs or expectations. These could only be identified through an evolutionary, collaborative process. Therefore, Agile was chosen as the better suited approach.

3.1.1 Practices and Artefacts

The development was done in timeboxed iterations, typically lasting three to four weeks. Requirements evolved throughout the project lifecycle, particularly due to strong user engagement. Regular meetings were held where progress was discussed and any change requests brought up.

Change management was implemented, in the form of following a number of steps: feasibility evaluation, discussion with the Computer Science team and the project supervisors, and implementation of required changes in the project plan (where appropriate).

Artefacts were only produced when necessary, to aid communication and understanding.

3.2 Requirements Gathering and Analysis Practices

A successful project is heavily reliant on understanding three key aspects: the stakeholders, the "As-Is" system, and the problems associated with it which the project aims to solve [26]. A three-hour workshop was held for this purpose, the results of which were subsequently analysed and consolidated (Sections 3.3 and 3.4).

Successively, requirements for the "To-Be" system (Section 3.5) were identified and refined, using interviews and observation.

Interviews were a standing item in the agenda of the regular meetings throughout the project duration, although lengthier and more prominent in early iterations. Open-ended questions were used mostly towards the early stages, while an increasing number of closed-ended questions were used in the later stages.

Observation was used for directly witnessing the user activities. This was performed both during Fastbleep Schools team meetings – where organizational issues were discussed – and during school sessions – an opportunity to communicate directly with the teachers.

3.3 Stakeholder Analysis

Stakeholder needs describe what stakeholders require for a successful project [27]. Understanding stakeholder needs is an important part of understanding the problem domain and therefore of creating a viable solution.

Stakeholders are typically grouped in the following categories: users, sponsors, developers, authorities, customers [26]. Only three of those apply in this case: sponsors (Section 3.3.1), users (Section 3.3.2), and developers (Section 3.3.3). For each stakeholder group, different stakeholder needs have been identified.

3.3.1 Sponsors

Sponsors are generally business managers, shareholders, steering committee members, those stakeholders that do not directly use the system, but rather are interested in the business outcomes that the system influences [26].

The sponsors of this project are made up of the Fastbleep executive team. They are not directly involved in the day-to-day operations of Fastbleep Schools but they set its high-level direction and long-term strategy. Their main focus is rapid expansion so that the current momentum is not lost.

Stakeholder Needs

The new system needs to be functional, sustainable and scalable in order to align with the sponsors' plans for the future of Fastbleep Schools.

3.3.2 Users

The actual users of the system are of three types: administrators, teachers, volunteers. Pupils, as a separate user type, were also included in early conversations. However, their needs and requirements were seen as so entirely different from those of the rest of the users that a separate dedicated project would be warranted. It was therefore established that pupils as users would remain outside the scope of this project.

In broad terms, users are interested in how the tool can actually improve the way they work with or as part of Fastbleep Schools.

Fastbleep Schools Administrators

The administrators are the Fastbleep School core team. They are responsible for leading the division's day-to-day operations. They will use the system to oversee the management of school sessions.

Stakeholder Needs

The system needs to facilitate the administrators' tasks of organizing school sessions. The process needs to be streamlined.

Teachers

The teachers will use the system to request sessions and check the status of their requests.

Stakeholder Needs

The system needs to improve the way teachers request and review bookings.

Volunteers

The volunteers will use the system to sign up for new school sessions.

Stakeholder Needs

The system needs to allow volunteers to easily sign up and undertake the preparation of school sessions.

3.3.3 Developers

Developers are the people in charge of the maintenance and support of the tool once it has been deployed. Even though Fastbleep do not currently have full-time IT staff on board, they have on-going collaborations with various technically able people. Therefore, despite the changing nature of the team, there will always be personnel in charge of the IT tools.

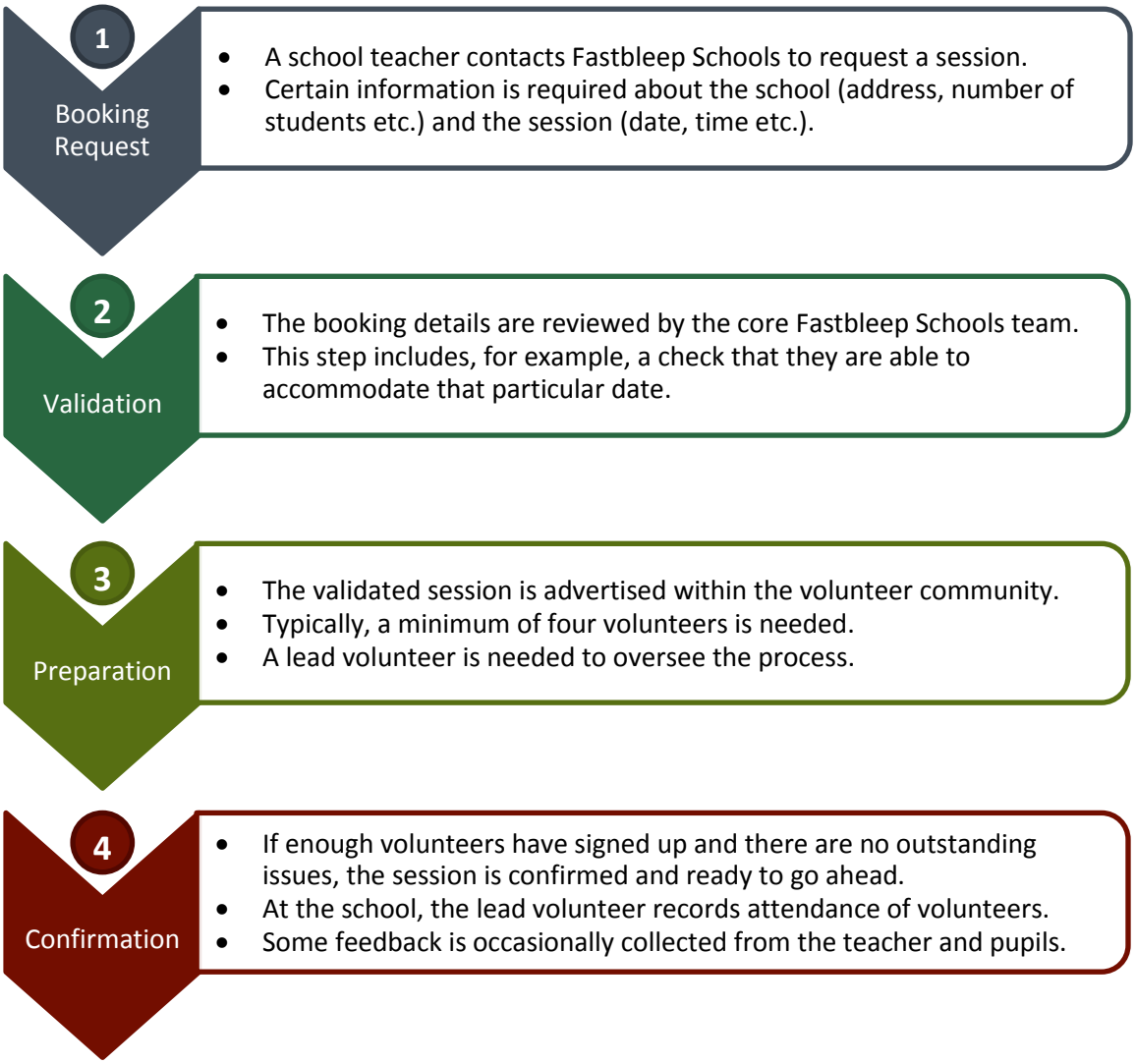
Stakeholder Needs

The system needs to be fairly simple, with a straightforward design, so that it can be easily understood and handed over to different people.

3.4 The "As-Is" System

The "As-Is" system is represented by the manner in which school sessions are currently requested and organized. Figure 3.1 outlines the steps typically involved in the process.

Figure 3.1: School Session Lifecycle



3.4.1 Tools and Technologies

A number of different tools and applications are utilized in each of the four steps of a school session lifecycle, in the absence of a unified system to perform these tasks. Table 3.1 outlines these tools and their specific uses.

Table 3.1: Typical Tools Utilized in a School Session Lifecycle

Step	Tool or Application	Description of Utilization
1. Booking request	FS website (Wordpress based)	Teacher fills in web form
	Podio (online work platform)	Form information stored in tables
2. Validation	Podio	FS administrators review session requests and manually copy validated ones to a different table
3. Preparation	Email, Facebook, Twitter, FS website	Advertise session among volunteers
	Email, Facebook, Eventbrite	Volunteers sign up
	Podio, Eventbrite	Keep track of assigned volunteers
	Email, Google Docs	Communicate with volunteers
4. Confirmation	Evernote	Record additional session details
	Google Docs, paper forms	Record session feedback

3.4.2 Major Issues

The previous section illustrates the unnecessary complexity associated with the current way of operating. An analysis, jointly carried out with the Fastbleep Schools team, has led to the identification of the major issues. These can be grouped in two broad categories: what is wrong with the current method of operating (current problems) and what is missing from it (lacking capabilities). Both categories of issues are outlined below.

Current Problems

Inefficiency

Fastbleep is a non-profit organization, meaning that all members freely dedicate their time and efforts to its activities. It is clear, therefore, that minimizing waste and inefficiency is crucial, particularly in times of scarce resources.

A review of current operations, however, has highlighted that administrators spend approximately 10 hours performing organizational tasks for every school session. This is largely owed to the number of tools utilized and the amount of manual work involved. Additionally, volunteers are discouraged from signing up as leads for similar reasons. A lead volunteer would typically spend a couple of hours per session executing basic tasks such as collating information and contact details for every volunteer. Difficulty is increased when, for example, a volunteer drops out and the lead must arbitrarily contact other students in search for a replacement.

Another issue with regards to inefficiency is related to schools that have an on-going collaboration with Fastbleep. The process of organizing a session for such a school is repeated every time, despite some of the information being already there from previous sessions. For example, a teacher has to fill out the form with the school information (e.g. address) every time they request a new booking.

Overall, this is a considerable problem not only because of ineffective usage of resources, but also because it impedes growth. Fastbleep Schools would be unable to accommodate more than one session per week, for example, under current circumstances, despite existing demand.

Frequent Errors

Whenever such an amount of manual work is involved, frequent errors inevitably occur. For example, one of the administrators can validate a booking request without noticing that particular timeslot had already been booked. Many errors also occur when information is manually copied, particularly from one platform to another.

Confusion often arises in regards to the volunteers signing up; because there is no automated confirmation of their registration, they often forget about it or drop out altogether.

The consequences are, evidently, quite serious: the resulting mix-ups can lead to sessions being amended or cancelled shortly before their planned timeslot, dissatisfied teachers and pupils, and the Fastbleep core team repeatedly having to jump in and fix the situation.

Lacking Capabilities

Storage of historical data

Both for internal and external reporting, Fastbleep need to keep track of progress over time, by storing information regarding members, schools, and sessions. For a non-profit organization, statistics such as number of sessions per term, number of active volunteers, and geographical spread of schools are essential for obtaining funding.

Currently, most of this data is either scattered in different places, partially captured or not stored at all. Any reports regarding the activity of Fastbleep Schools are completed by the core team members, as needed, on an ad-hoc basis. This has proven to be a difficult and extremely time-consuming exercise.

Additionally, having an integrated source for all their past data would provide the core team with a high-level view of Fastbleep's evolution over time and allow for informed decisions regarding future direction and improvement.

Capturing feedback

In order to improve and customize sessions, Fastbleep need a systematic way to gather feedback from teachers and pupils. Currently, this is only done informally or not at all.

This problem is partially solved by one of the Computer Science projects, which uses Q-Methodology to measure the impact of the interventions on pupils [28]. Teachers, however, have little input in the design of the sessions. In the case of those that decide not to request a second visit from the Fastbleep Schools team, it is unknown whether the reason was dissatisfaction with the session. If so, what measures should be taken to prevent this from happening again?

Finally, in the absence of a formal feedback mechanism, Fastbleep have been limited in what claims they could make regarding the positive impact and response of their sessions. Without official recommendations from teachers, funding is, once again, difficult to obtain.

Encouraging participation and accountability

One of the organization's key strategic aims is to foster communication and teamwork between people from different disciplines and backgrounds in order to come up with innovative ideas and solutions. However, within Fastbleep Schools, this is not truly achieved as there is a sense of fragmentation rather than community. Outside the regular open meetings, people do not feel part of the same organization and rarely communicate. This has a major impact on the activities and operations of Fastbleep Schools.

In the case of volunteers, there are high dropout rates as they do not feel a sense of accountability or commitment. One volunteer has explained that it is difficult to take something very seriously when they register via Facebook, which has an informal feel about it. Additionally, they are

quickly put off by the sense of disorganization when they receive large amounts of email messages and notifications via social networks.

Altogether, the lack of a common tool that all members working with Fastbleep Schools could access and use is hindering Fastbleep from developing as a united community.

3.5 The "To-Be" System

Requirements are capabilities and conditions to which the system – and more broadly, the project – must conform [8].

This section illustrates these requirements, based on the analysis performed on the "As-Is" system.

3.5.1 Use Cases and Functional Requirements

Six use cases were initially identified:

- Teacher requests session
- Teacher cancels session
- Administrator responds to request
- Administrator cancels session
- Volunteer signs up for session
- Volunteer cancels attendance

A detailed description, containing primary and alternative flows, was developed for each use case. Examples can be found in Appendix A.

Ultimately these use cases were analysed and refined so that the detailed functional requirements could be identified. These are summarized in Table 3.2.

Table 3.2: Initial Functional Requirements

ID	User	Requirement	Priority
1	Teacher	Request Session	High
2		View Own Session	High
3		Edit Session Details	Medium
4		Cancel Session	Medium
5	Administrator	View Sessions	High
6		Review Session	High
7		Validate Session	High
8		Cancel Session	High
9		View Past Sessions	Medium
10		View Volunteer Attendance	Low
11	Volunteer	View New Sessions	High
12		View Own Upcoming Sessions	High
13		View Past Attended Sessions	Medium
14		Sign Up for Session	High
15		Drop Out of Session	Medium

Following further refinement, additional requirements were identified, as per Table 3.3. Requirements 16 – 20 illustrate the decision to create a login-based system.

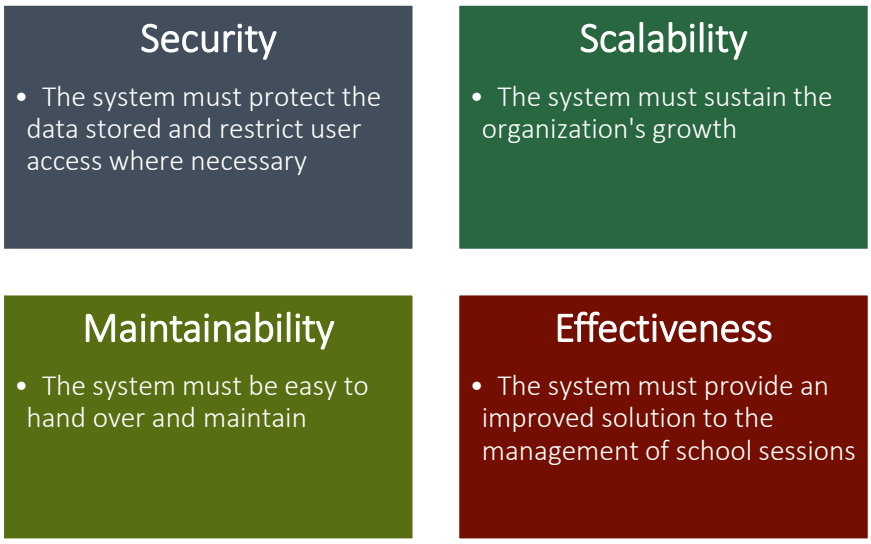
Table 3.3: Additional Functional Requirements

ID	User	Requirement	Priority
16	All	Register	High
17		Login	High
18		Change Password	High
19		Update Account	High
20		Delete Account	High
21	Administrator	Assign Lead	Medium
22		Submit Team Unavailability	Medium
23	Teacher	Change Schools	Medium
24		Leave Feedback	Medium
25	Volunteer	Sign Up as Lead	Medium
26		View Other Volunteers	Low

3.5.2 Non-Functional Requirements

A similar analysis has been carried out to identify all the non-functional requirements, summarized in Figure 3.2.

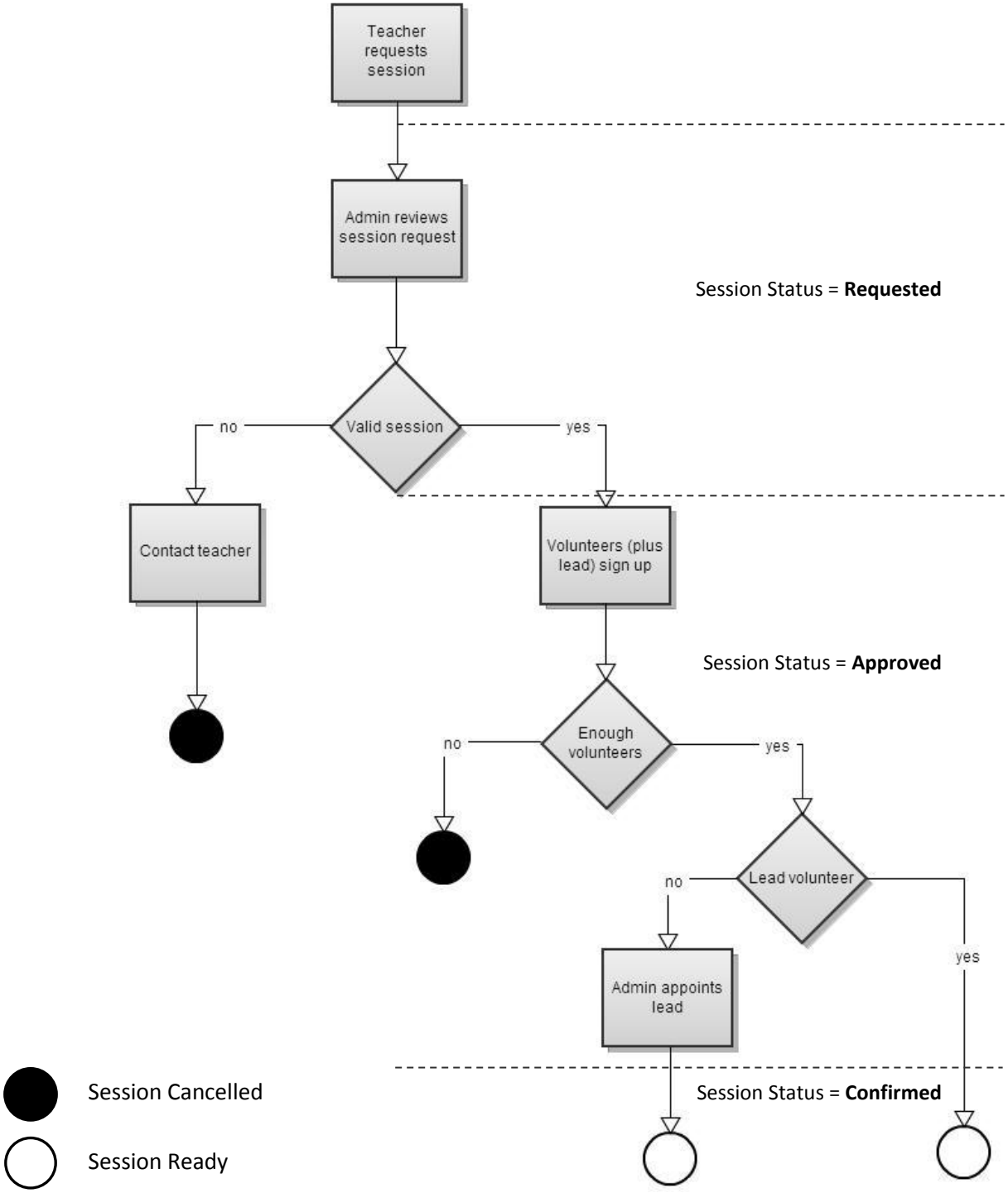
Figure 3.2 Non-Functional Requirements



3.5.3 Activity Flow

As a summary of how a school session will be managed in the new system, an activity flow diagram was created (Figure 3.3).

Figure 3.3: Activity Flow Diagram



3.6 Summary

Out of the two major development methodologies, Agile was chosen due to its suitability for this project. The practices associated with this approach (timeboxed iterations, continuous user engagement etc.) dictate how all the project activities are executed.

By comprehensively analysing the stakeholders and the "As-Is" system, the requirements and the process flow for the "To-Be" system were defined. The non-functional requirements were used in designing the application (Chapter 4), while the functional ones guided the implementation (Chapter 5). The activity flow helped in ensuring a thorough testing of the system (Chapter 7).

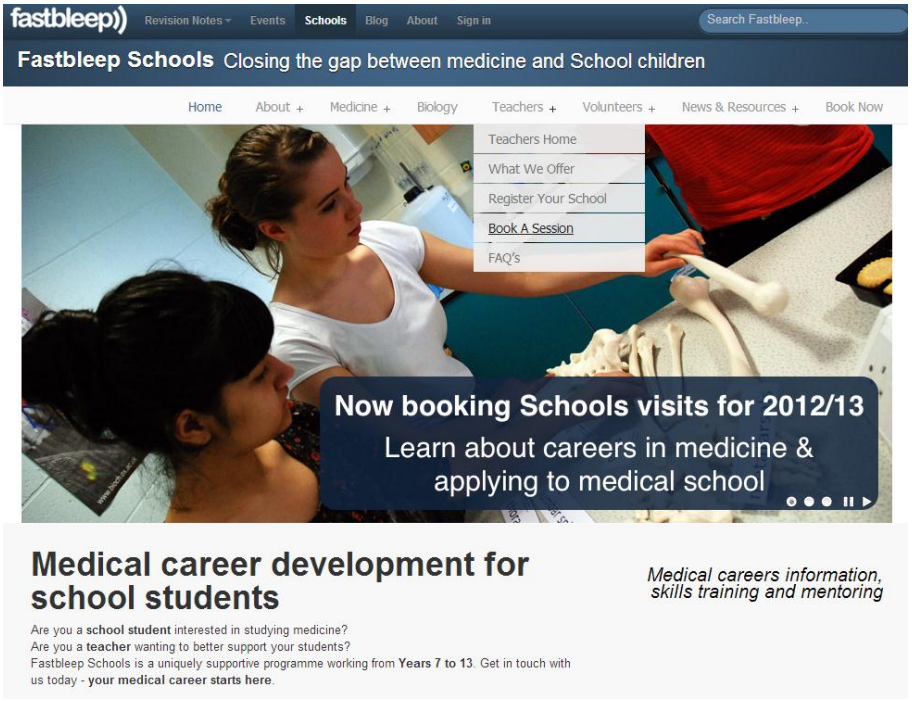
4 Design

This chapter initially presents the current Fastbleep Schools website and its functionality (Section 4.1), especially given the fact that the new system must be integrated within it. The design (Section 4.2) and architecture (Section 4.3) of the solution are then described, including reasoning behind some key decisions. Finally, a partial design class diagram is illustrated (Section 4.4) in order to provide the reader with a conceptual view of the system.

4.1 Current Infrastructure

As explained in Section 3.4, the current website (http://www.fastbleep.com/schools/), partially illustrated in Figure 4.1, significantly lacks functionality and this is compensated by utilizing various other tools and platforms. This project aims to fill these gaps as well as provide much needed additional capabilities.

Figure 4.1: The Fastbleep Schools Home Page



The website is powered by Wordpress and makes use of various instruments and widgets offered by the online platform. However, there is little functionality beyond the basics and the user-friendly interface.

There are web forms that handle, for example, booking requests and send the data into the Fastbleep Schools Tables on Podio. However, there is very little validation or manipulation of data. Therefore, a high number of duplicates and user errors are contained, which make the administrators' work even more difficult. There is no means for viewing upcoming or past sessions and the volunteers *Sign Up* page simply displays an e-mail address that they must contact.

In order to be integrated into this website, the solution delivered by this project will have to come in the form of a web application. Its pages will be incorporated within the website, replacing some of the current pages or simply being added, as necessary.

For seamless integration, an API created by the website's founder will be utilised. This API applies a series of CSS files to an application in order to transform its interface into the established Fastbleep format. The API provides the added benefit of allowing for the focus to remain on implementing back-end functionality rather than the front-end, which is the purpose of a different project (see Section 1.3).

4.2 Application Design

The design, both in terms of application structure and data model, underwent several iterations and continuous refinement. While the functional requirements played a key part in the concrete implementation of the tool, the non-functional requirements were fundamental in shaping the design.

Security

In order for the data stored to be protected against unwanted access and tailor the functionality to the different types of users, the decision was made to design a login-based system. According to the user role, different levels of restriction would be implemented. For instance, administrators would have more access rights than teachers and volunteers.

Scalability and Maintainability

In order to sustain potential restructuring, growth and easy maintenance, particular attention was given to the data model.

By freeing the schema from undesirable dependencies, duplication, redundancy and other such data anomalies, the application will be more likely to be of practical use to the organization over the long term. A more thorough description is included in Section 4.2.2.

Additionally, two key principles that guided the overall software design (partially illustrated in Section 4.4) were high cohesion and low coupling. They allow for a more understandable code that can be easily changed or built upon.

Effectiveness

In order for the application to effectively improve the management of school sessions, a “no frills” approach was adopted. This translated into a simple, clear, straightforward structure and design, instead of unnecessary features for the sake of an improved “look and feel” or user experience.

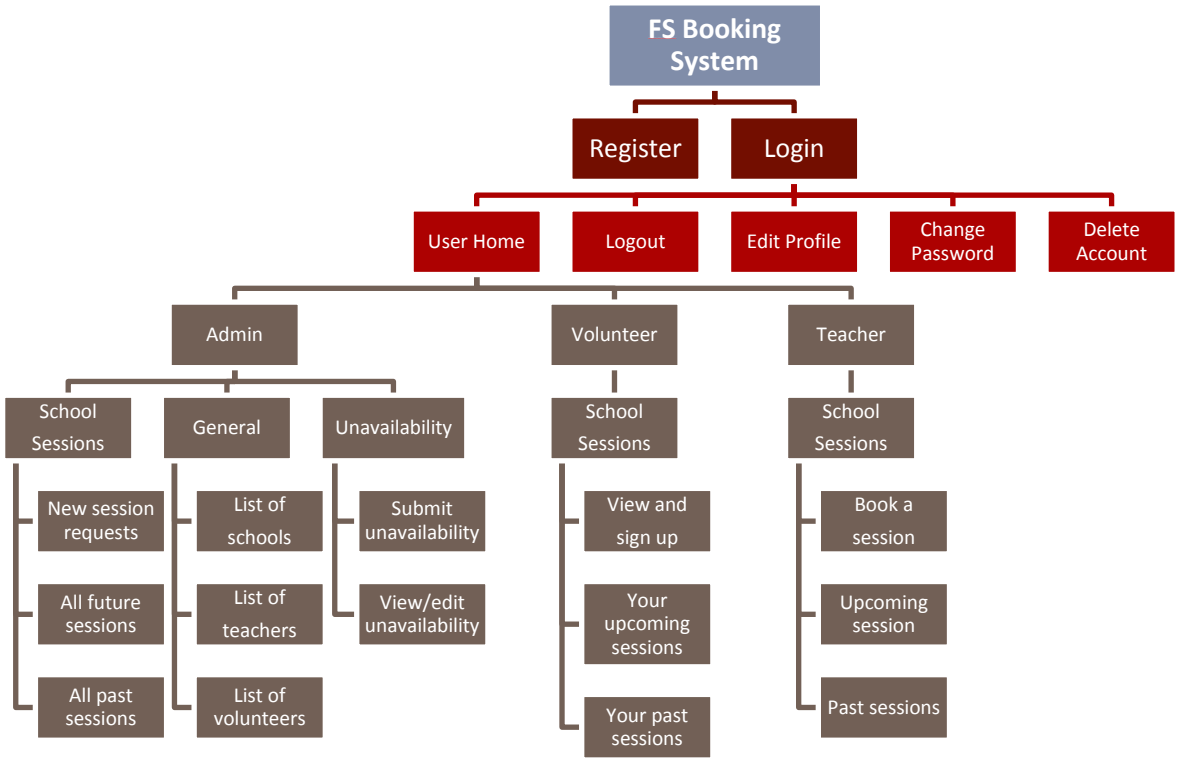
For example, administrators will be able to access a page where information regarding the upcoming sessions is consolidated into one table. This way, they are able to see immediately which sessions have not been approved yet, which are still missing volunteers etc., and they are able to act in a timely manner. The more user-friendly alternative would probably have been to display them one at a time, but that would have slowed down the administrator’s task and therefore missed the point.

The structure of the application is presented in Section 4.2.1.

4.2.1 *Application Structure*

The site map in Figure 4.2 illustrates how the application is virtually structured.

Figure 4.2: The Fastbleep Schools Booking System Site Map



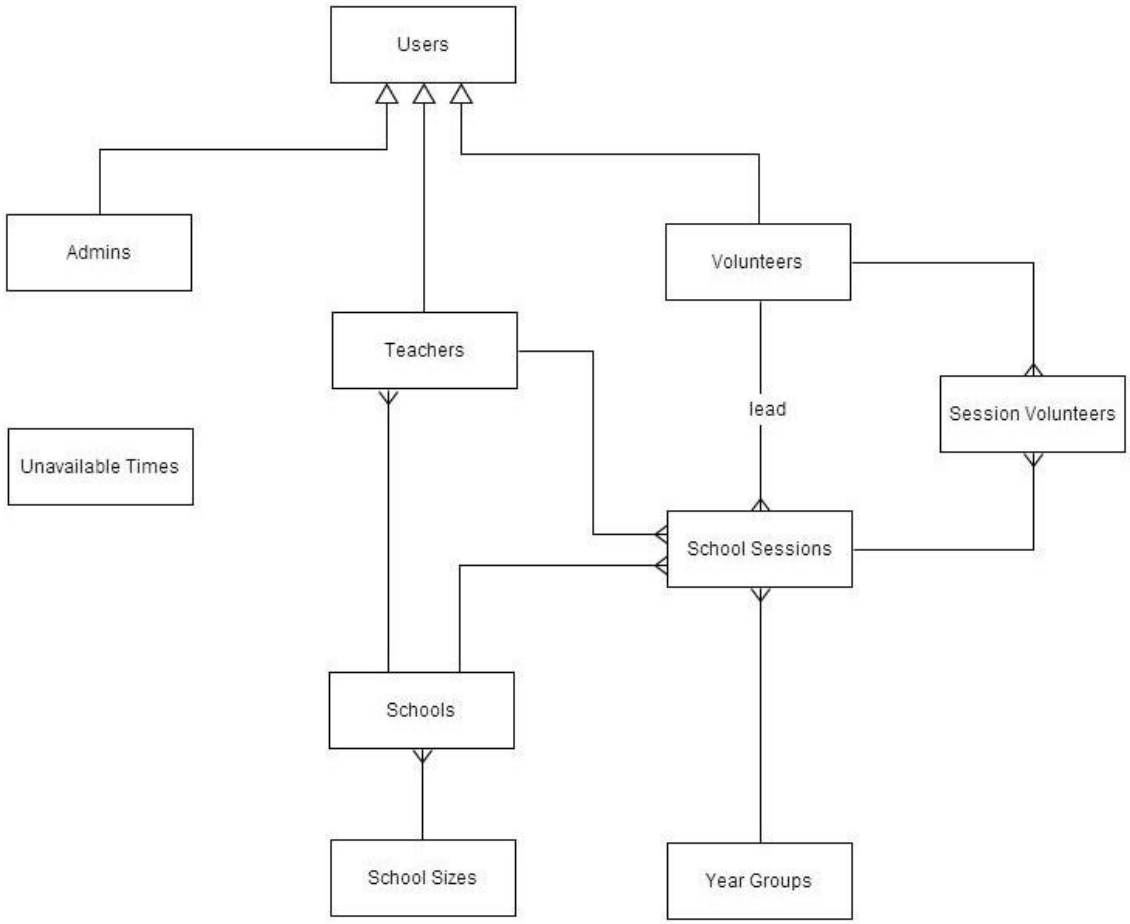
Once logged in, a customized home page and a different set of pages are displayed for each type of user.

When not logged in, the user will be able to visualize the existing contents of the Fastbleep Schools website. However, in order to access any of the new functionality, they will have to create an account. This decision was debated with the stakeholders. Initially, it was believed that a username and password will represent a hassle and that it will hinder people. Nevertheless, a brief survey with a few users has shown that they would much rather use their e-mail address and have a password-protected individual profile than continue accessing the system in an ad-hoc, disorganized manner. Additionally, some of the information stored (such as session details, pupils ages, date of birth etc.) was not considered appropriate for viewing by simply anyone who accessed the website.

4.2.2 Data Model

As per Agile practices [8], artefacts were not developed in great detail. A high-level conceptual data model was drawn and updated as it evolved from one iteration to the next. Figure 4.3 depicts the final version, as it stood at the time of project completion.

Figure 4.3: High-level Data Model



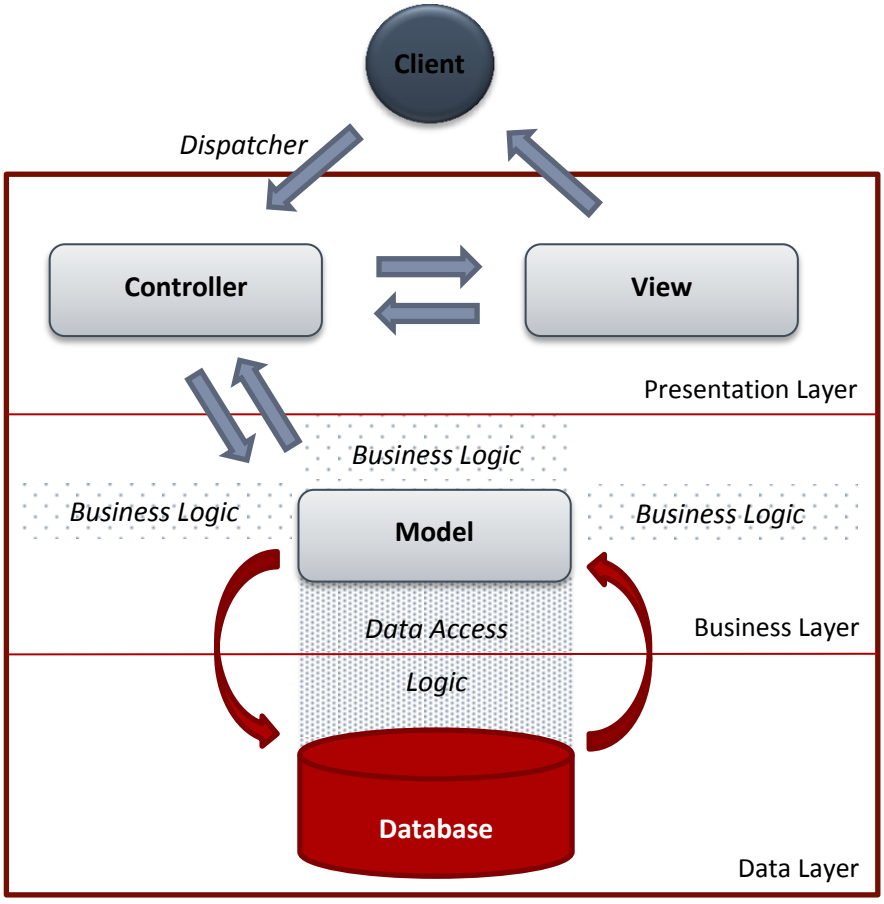
A few aspects to note are:

- The three different user types were modelled using a superclass-subclass relationship ("IS-A") because they have a set of common attributes (e-mail address, password, date of birth etc.) and a set of different attributes. They are also disjoint groups.
- Multiple teachers are allowed per school. It has been the case in the past for different teachers within one school to request sessions resulting in duplication of data. This would be eliminated by a many-to-one relationship.
- The *School Sizes* and *Year Groups* Tables have been added in view of potential extension of Fastbleep. For example, they currently only cater for a few age groups, for which they tailor their sessions; however that list might extend over time. Therefore, in order to avoid hard-coding any business-specific data and to keep the database schema normalized, these two tables have been added separately and not included in the *School Sessions* table.
- The many-to-many relationship between *Volunteers* and *School Sessions* is resolved via the *Session Volunteers* Table.
- The *Unavailable Times* Table has been added to allow administrators to submit dates or periods when the team is unable to accommodate sessions.

4.3 Application Architecture

The web application is built using the three-tiered architecture model and according to the Model-View-Controller paradigm [13]. The architecture of the application is illustrated in Figure 4.4.

Figure 4.4: Fastbleep Schools Booking System Architecture



When the client requests a page or resource in the application, a dispatcher directs it to the appropriate controller. The controller then communicates with the corresponding model and the request is fulfilled through the interaction between the model and the database. An example of a request would be retrieving a particular set of data. The output of the request is subsequently displayed via the appropriate view. The interactions between the different components are governed by data access and business logic (Fastbleep-specific).

4.4 Design Class Diagram

Finally, a key step in software design is system class modelling. This model evolved naturally, based on the data model and the application architecture.

Figure 4.5: Partial Design Class Diagram Illustrating the Controllers

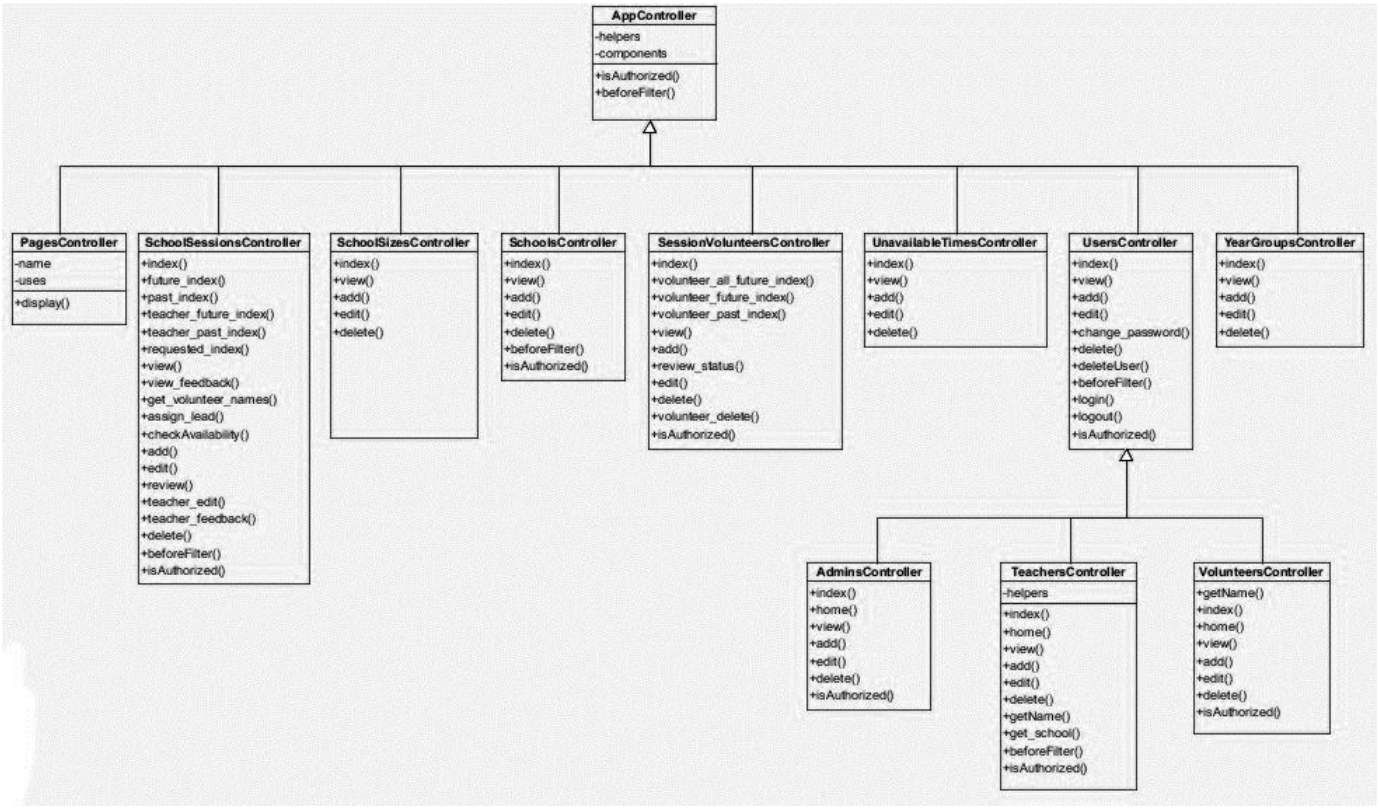


Figure 4.5 shows a partial diagram of the system classes, namely the controllers. The matching diagram for the models and the views would have a similar structure. Most of the functionality, however, is to be implemented in the controllers. Each of the methods illustrated uses models to retrieve and update data and has a corresponding view that handles user interaction.

In addition to the three types of classes, the application adopts typical conventions of the rapid development framework employed [14]. Components, elements, behaviours, helpers and library classes are used for various purposes, such as placing re-usable code or importing external libraries, vendors or APIs.

4.5 Summary

This chapter has looked at how the application was designed, both in terms of its structure and its data model, taking into account the current system infrastructure and the non-functional requirements.

The architecture and the software design highlight the use of the MVC pattern and constitute the foundation for the implementation of the system (Chapter 5).

5 Implementation

This chapter justifies the choice of tools and technologies used in the construction of the new Fastbleep Schools booking system (Section 5.1). Following a brief description of the implementation approach (Section 5.2), the key implementation aspects are illustrated, iteration by iteration (Section 5.3).

5.1 Development Tools and Technologies

For the client side implementation, HTML, CSS, and Javascript are tools commonly used in web applications. They were therefore chosen, to maintain simplicity and familiarity. Additionally, as mentioned in Section 4.1, an API using CSS would be used to alter the "look and feel" of the application so alternatives to CSS were not thoroughly considered.

In terms of server side implementation, PHP and MySQL were chosen for the following reasons:

- Wordpress websites (like the current Fastbleep Schools one) are written in PHP, use MySQL, and run on a Linux server platform.
- Prior knowledge and experience on the part of the author meant that development could start right away rather than potentially facing a time impediment due to learning something novel.
- They are extremely popular and widely accessible, so there is a high likelihood that anyone who will be in charge of the system in the future will be able to understand, maintain, and update it. A key non-functional requirement is therefore being met (maintainability; Section 3.5.2).
- A relational database is suitable for a project of this scale and complexity, as it allows for reliable access, querying and updates of data while maintaining fairly high performance levels [29].

An Apache web server was used to run PHP. This is also a widely-spread, free, open-source tool, commonly used in conjunction with PHP and MySQL.

JQuery was used for a few, particular cases where event handling and specific animations were necessary that could not be done using basic HTML.

The development was done in NetBeans IDE, given previous experience using this environment and the tool's helpful features for developing with PHP (e.g. auto completion, variable and class name recognition, code highlighting) [25].

5.1.1 Analysis of Rapid Development Frameworks

Three main rapid development frameworks for PHP were considered: Zend, Kohana, and CakePHP. They were selected based on generic knowledge and popularity among peers.

The analysis, reflected in Table 5.1, uses a RAG-type (Red, Amber, Green) rating system. Green represents a positive mark, Amber an average, while Red represents a negative one.

Five criteria were chosen for comparing the three frameworks (in no particular order):

1. **Learning difficulty.** Given no previous experience with any PHP rapid development framework, there would a learning curve in any case. However, the more difficult the tool, the more stringent time constraints would occur.
2. **Pervasiveness.** Once again, the widespread use of a tool could provide a significant benefit not only for its future maintainability in the context of Fastbleep Schools, but also as an addition to the author's CV and employability prospects.
3. **Documentation and user community.** Often, documentation and a large, active user community can represent a major support advantage for the developer, particularly in times of struggle or confusion.
4. **Features and plugins.** Application-specific logic often requires particular features or plugins that are not found in the standard implementation of a framework. The absence of such features could mean that the framework hinders development rather than enhancing it.

5 **Performance.** Some frameworks are known for slower page loads than others. Evidently, slow performance is not desired.

In light of the scoring of each criterion, the overall ratings reflect the perceived suitability of each framework for this project.

Key		
Red	Amber	Green

Table 5.1: Analysis of Rapid Development Frameworks**

Criteria	Zend	Kohana	CakePHP
Learning difficulty	Fairly high complexity, not considered suitable for beginners	Fairly clear-cut, but lacks a completed manual	Clear-cut; extensive manual; popular among peers
Pervasiveness	Extensively used in the industry	Not very widespread	Increasingly widespread
Documentation and user community	Large community, but quite passive; little support, few tutorials	Limited documentation, small and passive user community	Extensive documentation; large, active user community
Features and plugins	Fairly rich variety, but not all of them are free	Standard set of features and plugins	Great variety; numerous user-created sample plugins; all free
Performance	Average, slower than other frameworks	High performance, fast page loads	Slower than others, but improved performance in new version
Overall	Popular and powerful framework, but essentially unsuitable	Potentially suitable but the limited support poses a significant risk	There are some drawbacks, but seen as the most fitting

**the information included is broadly based on interviews with peers and views of the developer community, expressed on specialized forums (e.g. stackoverflow.com) and blogs.

The analysis has concluded that CakePHP is the best and most suitable rapid development framework for this project.

5.2 Implementation Approach

As discussed in Section 3.1, an Agile approach was chosen for the development of the application. From an implementation perspective, this translated into timeboxed iterations leading to the successive enlargement and refinement of the system (each iteration building upon the last) [8]. As part of this iterative lifecycle, regular meetings were held with the Fastbleep Schools core team. This approach has proven highly beneficial as it allowed for early and continuous user engagement and feedback.

Iterations have varied in length due to holidays and exam periods, but typically lasted for roughly three to four weeks.

5.3 Iterations and Key Implementation Aspects

In keeping with Agile practices [8], each iteration has included work in most disciplines, from requirements gathering to implementation to testing. However, the relative effort and the emphasis placed on each such activity have changed over time [8].

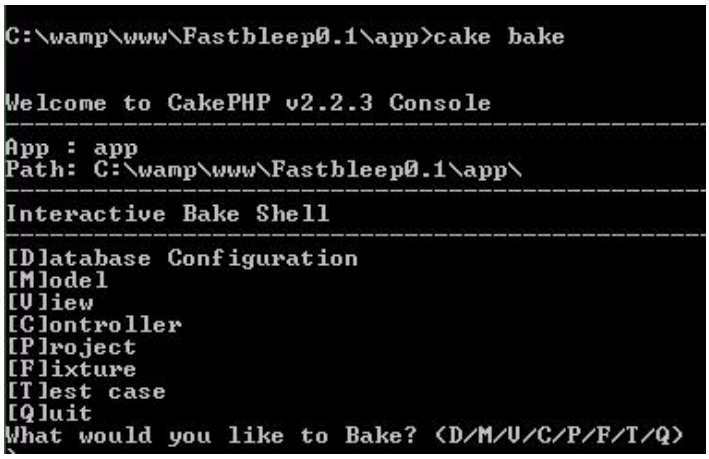
The main activities undertaken in the four iterations are outlined below, along with the key implementation aspects.

5.3.1 *Iteration 1: Code Generation with Bake*

The initial iteration was evidently focused on understanding and analysing user requirements. Domain and data models were drawn up to aid communication and facilitate understanding.

From an implementation perspective, an initial database schema was elaborated and executed in MySQL. Based on this, CakePHP was used for the automatic generation of models based on the database tables and of the corresponding templates for controllers and views. This feature is called "baking" in CakePHP and essentially entails running a script in order to set up early configuration and data validation criteria. Figure 5.1 partially shows the initial steps of the script.

Figure 5.1: The CakePHP Bake Script



This practice not only abides with the Agile methodology, which encourages early and visible progress, but also helped in speeding the learning process through hands-on experiments with CakePHP and its features. It also provided a starting point for future progress.

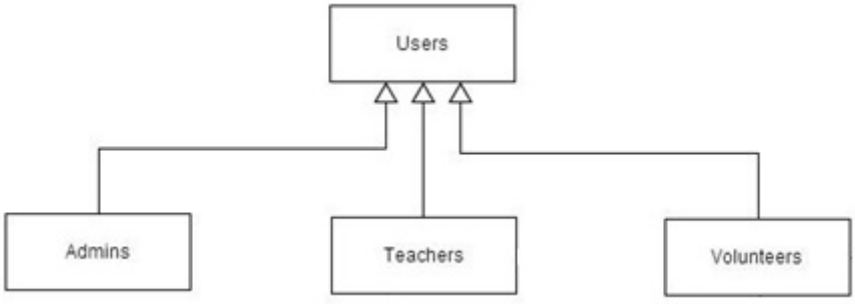
5.3.2 *Iteration 2: The Login System and Primary Functionality*

During the second iteration, requirements continued to be in focus, their analysis leading to a clearer and more refined view of the application's design.

The data model evolved accordingly and the database schema was altered to reflect the necessary changes. This time, only the new tables, associations and attributes were "baked".

Subsequently, the login system and the corresponding basic user functionality were implemented. As mentioned in Section 4.2.2 and highlighted in Figure 5.2, the different types of users were represented using specialization to define a superclass/subclass ("IS-A") relationship.

Figure 5.2: The Parent-Child Relationship for Modelling User Types



However, CakePHP does not support the implementation of this type of behaviour. Therefore, a few alterations were needed to achieve the needed functionality to represent the "IS-A" relationship. Firstly, the models were associated using the many-to-one relationship. Although inaccurate, this was necessary to achieve the required visibility between objects. Additionally, the controllers corresponding to the individual user types were implemented to extend the Users controller. That way, the "parent" controller acts as an interface while the "child" controllers override the respective methods with their specific functionality.

The views were implemented as reusable components, using CakePHP *elements* [14]. Elements are essentially mini-views: they contain blocks of presentation code that needs to be repeated on different pages of the web application. This way, the "parent" views could be embedded in the "child" views (e.g. the *Add User* functionality was implemented as an element and included in the *Add Teacher*, *Add Administrator* and *Add Volunteer* views).

This technique ultimately achieved the intended goal while supporting the design patterns of high cohesion and low coupling, to maintain a clean, clear code that is easy to maintain and build upon.

Subsequently, the High priority functionality was implemented (as per Tables 3.2 and 3.3 in Section 3.5.2).

5.3.3 Iteration 3: Secondary Functionality

By the third iteration, requirements had become more stable, although minor changes were still being made to the data model and to the application structure.

During this iteration, the agreed secondary functionality of the system was implemented (the Medium and Low priority requirements).

Additionally, significant thought went into improving the quality of the data inputted into the system by trying to minimize user errors where possible. Therefore, small bits of functionality were added to the application.

For example, a significant problem was that school data was duplicated in the old system. Now, by modelling and implementing a many-to-one relationship between teachers and schools, multiple teachers can sign up for the same school. When a teacher is trying to register, they have the option of selecting an existing school from a list generated from the database, or add a new one. This way the duplication of school information is eliminated.

Other examples include a default timeslot set by the administrators (either morning or afternoon) for sessions and the feature to submit team unavailability (e.g. during exam times). Little adjustments such as these can have a significant impact in preventing errors and wastefulness later in the process.

5.3.4 *Iteration 4: Authorization*

During the final iteration, a basic user interface was constructed, to facilitate demonstration and navigation between pages.

Additionally, the user authorization was implemented to restrict access where appropriate and prevent incorrect viewing or editing of sensitive or private information. This was achieved by overriding the AppController's *isAuthorized* and *beforeFilter* methods.

- *isAuthorized* is a method meant to check whether the user has the necessary permissions to access the requested resources [14]. It was used to implement the logic behind users being able to view, edit and delete only their own profiles and schools (in the case of teachers). Administrators were given more rights for viewing but are still not able to edit any information.
- *beforeFilter* is a method that is executed before any other action in the controller [14]. This was implemented in particular for school sessions, for evolving requirements in terms of user authorization. For example, only volunteers who have signed up for a session are able to view the attendance list.

By implementing these two methods, attempting to access resources inappropriately – even via URL manipulation (e.g. changing the ID in the URL in order to view or edit another user's profile) – would be impeded.

5.4 Summary

Following careful consideration and analysis of alternatives, the tools and technologies were chosen and the system was constructed, over four iterations.

The fully implemented application is thoroughly described in Chapter 6.

The iterative, incremental implementation was aimed at closely matching the functional requirements. Whether this has been achieved is the subject of Chapter 7, which outlines both how the system's functionality was tested and whether it has succeeded in meeting the user requirements.

6 System Walkthrough

A walkthrough of the application is illustrated in this chapter. Examples of features are presented for each user type: teachers (Section 6.1), administrators (Section 6.2), and volunteers (Section 6.3).

Additional functionality and features that are common to all users can be found in Appendix B.

For clarity, it is worth mentioning that the application will still be integrated into the current Fastbleep Schools website. However, the current owner of the website, who was responsible for completing a few necessary steps prior to deployment, had other commitments and was unable to deliver his part within the timeframe. Therefore, the full integration will be done at a later stage as part of the author's on-going engagement with the Fastbleep Schools team, but outside the scope of this project.

6.1 Teachers

A few of the important aspects of the teacher-related features are represented in the Figures below.

As mentioned in Section 5.3.3, when registering, teachers now have the option of choosing from a list of existing schools or add a new school. Depending on the option, the page contents dynamically change (Figures 6.1 and 6.2). This feature was implemented using JQuery methods for event handling.

Figure 6.1: Teacher Registration – Existing School

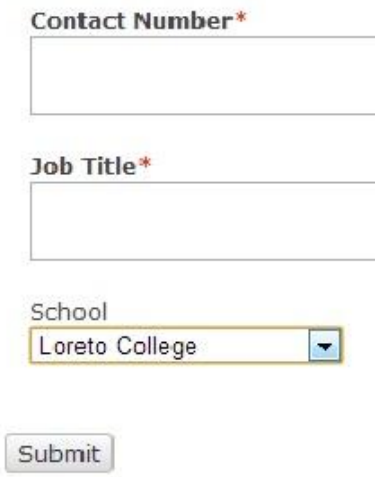
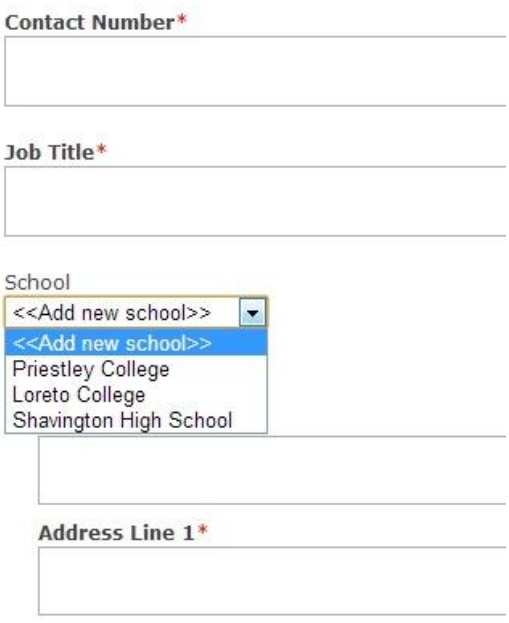


Figure 6.2: Teacher Registration – New School



When requesting a session, the input data is validated against form criteria and business rules. For example, the timeslot is checked against team availability (Figure 6.3). Once the request has been submitted successfully, the initial status of the session is set as *Requested*.

Figure 6.3: Unavailable Date for School Sessions

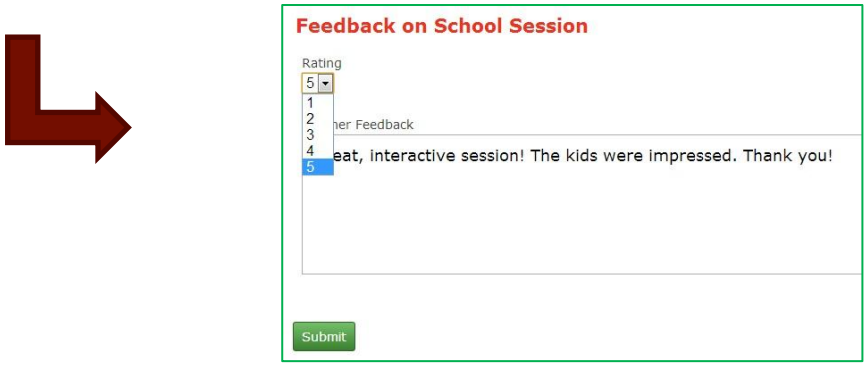
For upcoming sessions, teachers are able to track the status of sessions, edit session details and cancel a session. For the past sessions (Figure 6.4), teachers can submit feedback in the form of a rating from 1 to 5 and a text description (Figure 6.5).

Figure 6.4: Teacher's View of Their Past Sessions

Past school sessions at Loreto College

Date	Time Of Day	No Of Students	Year Group	No Of Volunteers Assigned	Lead Volunteer	Actions
Fri, 08 February 13	Morning	10	Years 12-13	4	Spencer Jim	Leave feedback
Fri, 15 March 13	Afternoon	10	Years 12-13	3	Sarah Jones	Leave feedback

Figure 6.5: Submitting Feedback

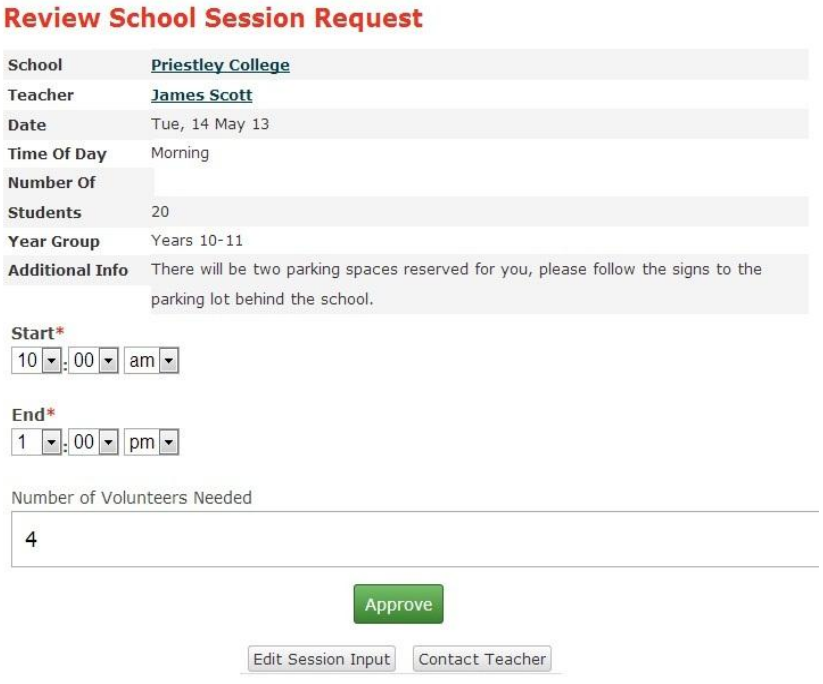


6.2 Administrators

Administrators can view new session requests, all future and all past sessions; they can also access the stored lists of teachers, volunteers, and schools; finally, they can submit and edit team availability.

Before a session can be opened for volunteer sign ups, an administrator must review it first (Figure 6.6). If, for some reason, the request cannot be met, the administrator can contact the teacher for further information or to simply let them know of the outcome. If the session is validated, its status is set to *Approved* and it becomes visible to volunteers for registration.

Figure 6.6: Administrator Page for Reviewing Sessions



For the validated school sessions, administrators can track progress by seeing their status, the number of volunteers who have signed up and whether there is a lead volunteer. Where there isn't a lead yet, the administrator can assign one from the list of volunteers.

In terms of team availability, the administrator can submit a particular time of the day, a day or a period that will be marked as unavailable for school sessions, as illustrated in Figures 6.7 and 6.8. When the period involves several days, the option for a time of day is disabled (Figure 6.8).

Figure 6.7: Administrator's View of Team Unavailability

Figure 6.8: Submitting Unavailability

Administrators can also view records of past sessions, the attendance and the feedback received from teachers (if any).

6.3 Volunteers

Volunteers can sign up for school sessions and view their upcoming sessions, as well as the ones they attended in the past.

Only those sessions that have been previously reviewed and validated by administrators will be visible to the volunteers for sign up. The option of offering to be the volunteer lead is also available if there is currently no lead assigned; Figures 6.9 and 6.10 illustrate the two situations.

Figure 6.9: Volunteer Sign Up – No Lead Yet

Figure 6.10: Volunteer Sign Up – Existing Lead

When the required number of volunteers has been reached and there is a lead, the session is set as *Confirmed*.

After signing up, volunteers can view their upcoming sessions and also drop out, if necessary. If the session had been confirmed and a volunteer drops out or if the lead volunteer drops out, the session is re-opened for registration (its status goes back to *Approved*).

Volunteers are also able to view a record of the sessions they have attended, along with the total number of hours of volunteering. This contributes towards their chances to win social impact awards for their work.

6.4 Summary

This chapter has illustrated a few of the main features of the system, by user type, while also describing the evolution of a school session through the three states (*Requested*, *Approved*, *Confirmed*).

The application is simple, yet flexible and straightforward. It can be easily extended to include additional functionality or more user types (e.g. pupils).

The full functionality of the system has been tested and evaluated against user requirements, as the next chapter will outline.

7 Testing and Evaluation

This chapter covers the testing activities performed over the project lifecycle (Section 7.1). Subsequently, the matter of whether the application ultimately met the initial objectives and the functional and non-functional requirements is evaluated (Section 7.2).

7.1 Testing

Software testing is an essential activity in assessing the quality of an application [30].

Testing activities can be classified in relation to two criteria: opacity and level of specificity. According to the former, there are two major categories: white-box and black-box testing. According to the latter, there are four types (from the lowest to the highest level): unit, integration, system, and acceptance testing. There is a typical correlation between the two classifications, in that white-box testing is usually associated with lower levels of specificity (unit, integration) while black-box testing with higher levels (integration, system, acceptance) [30].

In the case of the Fastbleep Schools booking system, white-box testing was performed by the author throughout the lifecycle, in each iteration, as per Agile practices [8]. Black-box testing, on the other hand, because of its very nature, was particularly accentuated towards the final iterations. The next two sections outline how both these activities were undertaken.

7.1.1 White-Box Testing

White-box testing, or structural testing, takes into account the internal mechanism of a system or component [31] and is typically used to verify whether or not the code works as expected [32].

This type of testing activity was performed mostly in the form of integration testing – evaluating the interaction between the different software components [31]. This effectively translates into whether or not the different controllers and their functions work well together in the end-to-end manipulation of school session data. There are several techniques to achieve this evaluation, but the one chosen is path testing (outlined below) [32].

CakePHP also proved to be useful for white-box testing, through a log that displays information on the database queries executed for a particular request. Figure 7.1 shows an example of such a log, which includes the actual queries, the number of records returned and/or updated, time taken to execute the queries and the errors caused (if any). This feature not only enabled effective testing of the database interaction, but also facilitated fast and easy debugging.

Figure 7.1: The CakePHP Query Log

(default) 2 queries took 15 ms

Nr	Query	Error	Affected	Num. rows	Took (ms)
1	SELECT `Volunteer`.`user_id`, `Volunteer`.`degree`, `Volunteer`.`graduation_year`, `Volunteer`.`mobile_number`, `Volunteer`.`CRB_reference_no`, `Volunteer`.`date_of_birth`, `Volunteer`.`suture`, `Volunteer`.`driver`, `Volunteer`.`attended_training`, `User`.`id`, `User`.`role`, `User`.`name`, `User`.`email`, `User`.`password` FROM `fastbleep_schools1`.`volunteers` AS `Volunteer` LEFT JOIN `fastbleep_schools1`.`users` AS `User` ON (`Volunteer`.`user_id` = `User`.`id`) WHERE 1 = 1 LIMIT 20		4	4	14
2	SELECT COUNT(*) AS `count` FROM `fastbleep_schools1`.`volunteers` AS `Volunteer` LEFT JOIN `fastbleep_schools1`.`users` AS `User` ON (`Volunteer`.`user_id` = `User`.`id`) WHERE 1 = 1		1	1	1

Path Testing

Path testing is designed in such a way to execute all or selected paths through a system [31]. For this purpose, the activity flow diagram in Figure 3.3 (Section 3.5.3) was used. The four paths identified were numbered from left to right and tested using test data and scenarios. The results are summarized in Table 7.1.

Table 7.1: Path Testing and Results

Path ID	Path / Test Description	Test Outcome	Result
1	Administrator considers that the request cannot be fulfilled and decides to contact the teacher.	Administrator is led to a page displaying the teacher's contact details.	Passed
2	The request has been validated. The session date is approaching but not enough volunteers have signed up.	Session status still appears as "Approved". Administrator tracks it up to three days prior to the planned date and cancels it. The session is removed.	Passed
3	The required number of volunteers has signed up to a given session but no one has offered to lead it.	Administrator views the list of volunteers and appoints one of them as the lead. The session is set as "Confirmed".	Passed
4	The required number of volunteers has signed up and one of them has offered to lead it.	The session is set as "Confirmed" and appears on each of the volunteers' attendance records. The lead is able to view the list of volunteers and their contact details.	Passed

7.1.2 Black-Box Testing

Black box testing, or functional testing, ignores the internal mechanism of a system or component and focuses solely on the outputs generated in response to selected inputs and execution conditions [31].

This type of testing was performed at two levels: system testing (by the author) and acceptance testing (with the help of the Fastbleep Schools team).

System Testing

System testing is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements [31].

In addition to testing the functionality for all the agreed requirements, system testing involves putting the new application in different environments to ensure the program works in typical customer environments with various web browsers [30].

The five most popular web browsers (in their latest versions) were used to perform this evaluation [33] Appendix C contains screenshots illustrating a sample view represented in the chosen browsers. The evaluation has concluded that the application demonstrates little to no difference when using different browsers.

Since system testing is done with a full system implementation, it was performed during the final iteration.

Acceptance Testing

Acceptance testing is generally conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to decide if they would accept the system [31].

Acceptance testing was carried out over the course of three sessions attended by one of the sponsors (a Fastbleep executive), the Fastbleep Schools team, a representative of the volunteer community, and a developer (the founder of the current website). The points raised and discussed are summarized below.

- ✓ The system has the potential to significantly improve the way of working for all users. The time spent on minor organizational tasks is radically reduced.
- ✓ The overall user experience is enhanced, comparatively to the old system.
- ✓ Due to following the MVC pattern, as well as high cohesion and low coupling, the developer is content with being handed over the support and maintenance of the system.
- ✓ The system complies with the organization's long-term goals by being able to sustain increased demand.
- ☒ The system should facilitate more interaction and communication between users; a closer resemblance to a social network would be preferred.

Similarly to system testing, acceptance testing was performed during the final iterations.

7.2 Evaluation

In order to evaluate the overall success of the project, the initial objectives are referred to (Section 1.4).

I. Ensure that user requirements and expectations are met to the highest possible extent

All stakeholders (especially users) were closely involved in the development of the project. In the final sessions, they declared that the system does indeed reasonably match their evolving requirements and expectations.

II. Increase efficiency of the management of schools sessions

The system has the potential to substantially reduce the time wasted on organizational tasks. The additional features improve the user experience and in no way increase the overhead.

III. Build something that is easy to use and maintain

The system has been designed with the key non-functional requirements in mind. Therefore, it is fairly straightforward and thus maintainable.

IV. Build something that can be integrated within current systems

The system can be divided into different components that can then be integrated within the current website. Prior to that, however, the current website needs to be re-organized.

7.2.1 Achievements

On the whole, the project has met the required objectives to a reasonable extent, given the timeframe, the high degree of uncertainty, the complex and varied stakeholder needs, and the learning curve involved for the author. According to the results recorded in formal and informal feedback sessions, the stakeholders are satisfied with the final product.

In terms of the collaboration between Fastbleep and the School of Computer Science, an indicator of the successful results is the fact that they will be presented at an international conference. The Association for Medical Education in Europe (AMEE) holds an annual conference on teaching, research, and learning skills in healthcare. An abstract has been submitted and subsequently accepted, outlining how working across disciplines (medical education and computer

science) has been beneficial to widening access to medicine. The abstract can be found in Appendix D.

A poster will be prepared for presentation at the conference, the purpose being to stimulate a discussion on the topic, as part of this year's theme – "Colouring Outside the Lines" [34]. The theme refers to challenging "preconceived ideas" and encourages thinking of "new ways of working to produce future healthcare professionals to meet the needs of society in these times of limited resources" .

7.2.2 Deficiencies

Nevertheless, a critical analysis reveals that the application still lacks a series of features that, although labelled as "low priority", are considered of high value to the users. An example is represented by the resemblance with a social network mentioned above. Another would be the implementation of a notification and/or confirmation system, either via the user profiles on the website (similarly to Facebook) or via e-mails.

Additionally, the difficulty in a project with "real-life" users is that its benefits can only be proven through practical usability over a long period of time. Although it speeds up the whole process in test scenarios, the situation might be different in reality. Seeing that the application has not yet been deployed, this is extremely difficult to assess.

8 Conclusions and Next Steps

This chapter concludes the report, by presenting the outcomes of a reflection over the project (Section 8.1), a brief summary (Section 8.2), and future plans (Section 8.3).

8.1 Reflection

Following a relatively in-depth reflection of the project as a whole, its development and its results, a few noteworthy points can be made regarding the challenges faced (Section 8.1.1) and the lessons learned (Section 8.1.2).

8.1.1 Challenges

"Real life" Users

Despite being one of the reasons for choosing this project, working with actual users from outside the School of Computer Science, has proven to be the most challenging aspect. The requirements gathering and analysis activities were far more complex and time-consuming than initially envisaged. Additionally, managing expectations while responding to an increasing number of requests from the part of the stakeholders proved to be an extremely demanding task.

Ambiguity

Seeing the needs of the Fastbleep organization had changed from April to September 2012, the projects initially assigned to the Computer Science team no longer existed at the start of term. Therefore, a significant delay was caused by attempting to identify areas where Fastbleep could benefit most from IT help while corresponding to the individual skill sets and aims of the CS students. Additionally, the projects ultimately agreed upon have still changed direction later on as requirements were being refined and more analysis was done.

8.1.2 Lessons Learned

"Real life" Experience

Probably the most valuable lesson learned stems from working with an actual organization. In addition to technical skills, the ability to apply theoretical knowledge (e.g. Agile procedures) in practice was essential. Although the system and the requirements stabilized over time, they were never actually "locked"; implementation of additional features, as requested by users, was done throughout the four iterations. An additional lesson is the difficulty involved in building a common tool that fulfils the requirements of several groups of stakeholders, who have distinct interests and views.

Full Project Lifecycle

Experiencing a full project lifecycle, with all the typical activities, from requirements to implementation and testing, also proved to be highly constructive. The major benefit was the opportunity to perform different roles within the software development process (project manager, business analyst, developer, tester).

Technical Skills

The technical skills gained are valuable as well. The MVC pattern and rapid development framework concepts are both highly utilized in the industry and were useful to acquire, particularly considering no prior experience or knowledge of either. Moreover, the necessity to learn them in such a short timespan represented a preparation for future employment, when a demonstration of fast, on-the-job learning of complex notions will be critical.

Change Management

A consequence of working with an actual organization and with university peers is learning how to accept and deal with change. There have been a number of major changes that influenced the direction of the project. The author's ability to understand and respond to those changes while undertaking the necessary measures to remain on track has significantly improved over time.

8.2 Report Summary

This project was aimed at providing the Fastbleep Schools initiative with a tool that would facilitate and improve their operations and their customer interaction. Given rapid growth and the absence of a robust IT system, the management of school sessions had become a time-consuming and resource-draining process.

An Agile approach was adopted for the development of the solution. The requirements were identified on the basis of close user involvement; they evolved and changed over the project lifecycle. In accordance, the design of the system was initially drafted and subsequently altered over the course of several iterations.

The implementation was done in incremental steps, each iteration building upon the previous one. A thorough analysis was carried out to identify the most suitable tools and technologies. Testing activities were undertaken regularly; different types and techniques were applied at different times, as appropriate.

The final result is widely believed to have met both the project objectives and the user requirements, despite the challenges faced and the restricted timeframe. Although perhaps still lacking in potentially useful features, the application provides a solid basis for further advancement. For example, the next version of the system that focuses more on the social aspects can be implemented by future IT employees at Fastbleep.

8.3 Next Steps

As previously mentioned, the author's engagement with the Fastbleep organization will continue over the following months. As part of this engagement, the tool will be prepared for integration (e.g. data migration) and deployment. An additional task will be overseeing the short-term post-deployment activities, in order to accurately assess the practical usability and the real benefits of the application. This will contribute to the lessons learned from this project as well as the author's personal development.

Bibliography

[1] Department for Employment and Learning, "Access to Success - An Integrated Regional Strategy for Widening Participation in Higher Education," Department for Employment and Learning, London, 2011.

[2] J. Giles, E. Hill, F. Liuzii and S. Vaughan, "Identifying barriers and evaluating support: How can The University of Manchester best engage underrepresented students?," Fastbleep, Manchester, 2012.

[3] British Medical Association, "Equality and diversity in UK medical schools," British Medical Association Marketing & Publications, London, 2009.

[4] T. Greenhalgh, K. Seyan and P. Boynton, ""Not a university type": focus group study of social class, ethnic, and sex differences in school pupils' perceptions about medical school," *British Medical Journal,* vol. 328, 2004.

[5] J. Mathers and J. Parry, "Why are there so few working class applicants to medical schools? Learning from the success stories," *Medical Education,* vol. 43, 2009.

[6] K. Seyan, T. Greenhalgh and D. Dorling, "The standardised admission ratio for measuring widening participation in medical schools: analysis of UK medical school admissions by ethnicity, socioeconomic status, and sex," *British Medical Journal,* vol. 328, 2004.

[7] Fastbleep, *Fastbleep - A New Future For Education,* Manchester: Fastbleep, 2012.

[8] C. Larman, Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, Prentice Hall, 2004.

[9] "Free On-Line Dictionary of Computing," 23 November 1997. [Online]. Available: http://foldoc.org/waterfall. [Accessed 16 April 2013].

[10] L. Shklar and R. Rosen, Web Application Architecture: Principles, protocols and practices, John Wiley & Sons, Ltd, 2003.

[11] "Intro to Web Application Functionality," 2011. [Online]. Available: http://bmstevens.com/blog/intro-to-web-application/. [Accessed 17 April 2013].

[12] Microsoft, Microsoft Application Architecture Guide, Microsoft Press, 2009.

[13] A. Leff and J. T. Rayfield, "Web-Application Development Using the Model/View/Controller Design Pattern," in *Enterprise Distributed Object Computing Conference Proceedings (Fifth IEEE International)*, 2001.

[14] Cake Software Foundation, "CakePHP Cookbook Documentation, Release 2.x," CakePHP, 2013. [Online]. Available: http://book.cakephp.org/2.0/_downloads/en/CakePHPCookbook.pdf. [Accessed 10 April 2013].

[15] W3 Schools, "HTML Introduction: What is HTML?," 2013. [Online]. Available: http://www.w3schools.com/html/html_intro.asp. [Accessed 18 April 2013].

[16] W3 Schools, "CSS Introduction: What is CSS?," 2013. [Online]. Available: http://www.w3schools.com/css/css_intro.asp. [Accessed 18 April 2013].

[17] W3 Schools, “Javascript Introduction: What is Javascript?,” 2013. [Online]. Available: http://www.w3schools.com/js/js_intro.asp. [Accessed 18 April 2013].

[18] W3 Schools, “PHP Introduction: What is PHP?,” 2013. [Online]. Available: http://www.w3schools.com/php/php_intro.asp. [Accessed 18 April 2013].

[19] W3 Schools, “MySQL Introduction: What is MySQL?,” 2013. [Online]. Available: http://www.w3schools.com/php/php_mysql_intro.asp. [Accessed 18 April 2013].

[20] “Apache HTTP Server Project,” 2012. [Online]. Available: http://httpd.apache.org/ABOUT_APACHE.html. [Accessed 21 April 2013].

[21] “Background. What is Apache?,” 2013. [Online]. Available: http://wiki.apache.org/httpd/FAQ#What_is_Apache.3F. [Accessed 21 April 2013].

[22] Zend Framework, “About Zend Framework 2,” Zend Framework, 2013. [Online]. Available: http://framework.zend.com/about/. [Accessed 10 April 2013].

[23] Kohana, “Kohana User Guide,” Kohana, 2012. [Online]. Available: http://kohanaframework.org/3.3/guide/kohana#what-is-kohana. [Accessed 10 April 2013].

[24] W3 Schools, “jQuery Introduction: What is jQuery?,” 2013. [Online]. Available: http://www.w3schools.com/jquery/jquery_intro.asp. [Accessed 18 April 2013].

[25] “NetBeans IDE Features - Overview,” NetBeans, 2013. [Online]. Available: https://netbeans.org/features/index.html. [Accessed 20 April 2013].

[26] K. Bittner and I. Spence, “Chapter 3: Establishing the Vision,” in *Use Case Modeling*, Pearson Education, 2003.

[27] S. McEwen, “Requirements: An Introduction,” IBM, 2004.

[28] Q. Ashraf, “Group based Q sort to support widening participation in medicine,” School of Computer Science, University of Manchester, 2013.

[29] D. Wesley, “Relational Database Design,” *Journal of Insurance Medicine,* vol. 32, no. 2, pp. 63-70, 2000.

[30] L. Williams, “Chapter 3: Testing Overview and Black-Box Testing Techniques,” in *A (Partial) Introduction to Software Engineering Practices and Methods*, ed. L. Williams, 2009.

[31] IEEE, “IEEE Standard Glossary of Software Engineering Terminology,” no. 610.12, 1990.

[32] L. Williams, “Chapter 4: White-Box Testing,” in *A (Partial) Introduction to Software Engineering Practices and Methods*, ed. L. Williams, 2009.

[33] W3 Schools, “Browser Statistics,” 2013. [Online]. Available: http://www.w3schools.com/browsers/browsers_stats.asp. [Accessed 23 April 2013].

[34] The Association for Medical Education in Europe (AMEE), “AMEE 2013: Colouring Outside the Lines,” AMEE, 2013. [Online]. Available: http://www.amee.org/index.asp?pg=292. [Accessed 29 April 2013].

Appendices

Appendix A: Examples of Use Case Descriptions

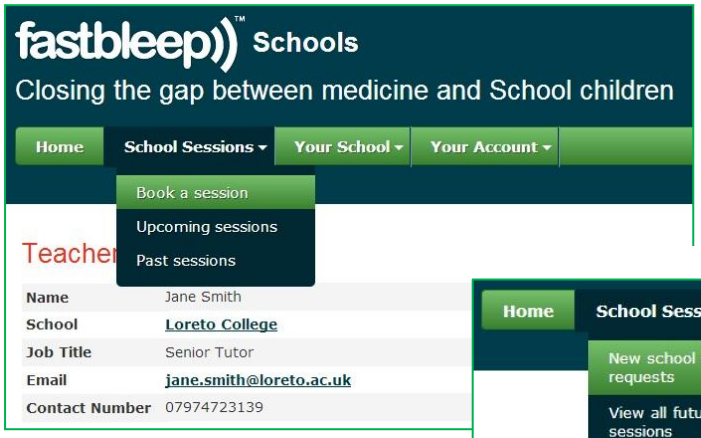
Use Case Name:	**Teacher Requests Session**
Scenario:	The teacher logs in and wants to book a School Session.
Triggering Event:	Teacher clicks "Book Now" for the Schools Session.
Brief Description:	Teacher selects the option to book a schools session, selects a date and inputs necessary information.
Actors:	Teacher
Related use cases:	Administrator Responds To Request, Teacher Cancels Session
Preconditions:	Teacher logged in.
Post conditions:	Request sent to Administrator for approval.
Flow of Activities:	1. Teacher clicks "Book Now" 2. Teacher selects date/time. 3. Teacher views and fills in form for event-specific information (number of pupils, age of pupils etc) 4. Teacher clicks "Submit Request" 5. Teacher receives notification – "Request Submitted Successfully."
Exception Conditions:	Teacher not logged in – pop up for Sign Up/Login Compulsory info in form not filled in – message (Validation)

Use Case Name:	**Administrator Responds To Request**
Scenario:	Administrator validates the session by checking details.
Triggering Event:	Teacher requests session.
Brief Description:	After teacher has requested a session, Administrator reviews the details. If details are correct and ok, Administrator validates the session.
Actors:	Administrator, Teacher
Related use cases:	Teacher Requests Session
Preconditions:	Administrator logged in Teacher raises request on system
Post conditions:	Request appears as "Approved" and is opened for next steps (volunteer sign ups)
Flow of Activities:	1. Administrator gets notification that teacher raised request (not always necessary –can just check the system to see if there are any pending requests) 2. Administrator clicks on "Review Request" 3. Administrator verifies request details 4. Administrator clicks on "Approve Request" 5. Request becomes "Approved" 6. Notification sent to teacher "Session planning underway!" 7. Session becomes open for volunteer sign ups etc
Exception Conditions:	Request pending for x number of days – reminder Request invalid – Administrator contacts the teacher Request incomplete – Administrator allowed to edit session details

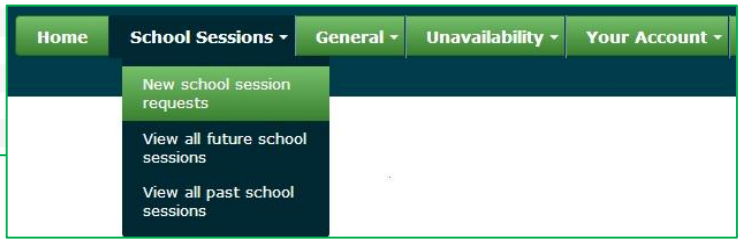
Appendix B: Further Screenshots of the System

Once logged in, users view a custom home page where they have the choice to navigate to different other pages. Each of these pages allows the user to perform various tasks, such as cancelling an upcoming session or leaving feedback for a past session.

Teacher Menu



Administrator Menu



Teacher's View of Upcoming Sessions

Upcoming school sessions at Priestley College

Date	Start	End	No Of Students	Year Group	Additional Info	Status	Actions	
Mon, 06 May 13	10:00:00	13:00:00	10	Years 12-13		Approved	Edit session details	Cancel session
Tue, 14 May 13	10:00:00	13:00:00	20	Years 10-11	There will be two parking spaces reserved for you, please follow the signs to the parking lot behind the school.	Requested	Edit session details	Cancel session

Page 1 of 1, showing 2 records out of 2 total, starting on record 1, ending on 2

< previous next >

Administrator's View of Upcoming Sessions

School Sessions

School	Teacher	Date	Time Of Day	Start	End	No Of Students	Year Group	Additional Info	No Of Volunteers Needed	No Of Volunteers Assigned	Lead Volunteer	Status	Actions		
Priestley College	James Scott	Mon, 06 May 13	Morning	10:00:00	13:00:00	10	Years 12-13		4	2	Sean Jeffrey	Approved		View	Delete
Priestley College	James Scott	Tue, 14	Morning	10:00:00	13:00:00	20	Years 10-11	There will be two parking	4	0		Requested	View	Assign Lead	Delete

Volunteer's View of Upcoming Sessions

Your upcoming school sessions

Date	School	Teacher	Start	End	No Of Students	Year Group	Additional Info	No Of Volunteers Needed	No Of Volunteers Assigned	Lead Volunteer	Status	Actions	
Tue, 14 May 13	Priestley College	James Scott	10:00:00	13:00:00	20	Years 10-11	There will be two parking spaces reserved for you, please follow the signs to the parking lot behind the school.	4	1	Spencer Jim	Approved	View	Drop out

Volunteer's View of Past Sessions

Your past school sessions

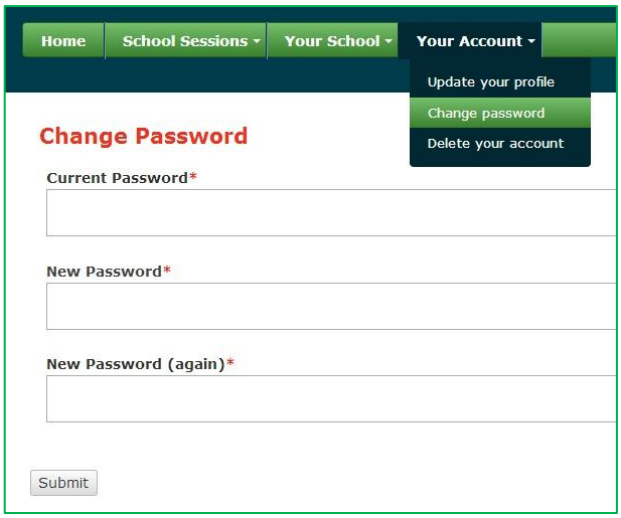
School	Teacher	Date	Start	End	No Of Students	Year Group	No Of Volunteers Assigned	Lead Volunteer	Rating	Actions
Shavington High School	Jason Shane	Mon, 18 March 13	10:00:00	13:00:00	20	Years 10-11	4	Spencer Jim		View
Priestley College	James Scott	Mon, 11 March 13	10:00:00	13:00:00	15	Years 12-13	4	Simone Jensen	5	View
Loreto College	Jane Smith	Fri, 08 February 13	10:00:00	13:00:00	10	Years 12-13	4	Spencer Jim	5	View

Total number of hours volunteered: 9

A common login system was created for all users using the parent-child relationship between the *User* class and the classes representing the different user types. Therefore, common functionality was implemented in the "parent" class. This involves the typical user features, such as editing the profile and changing the password.

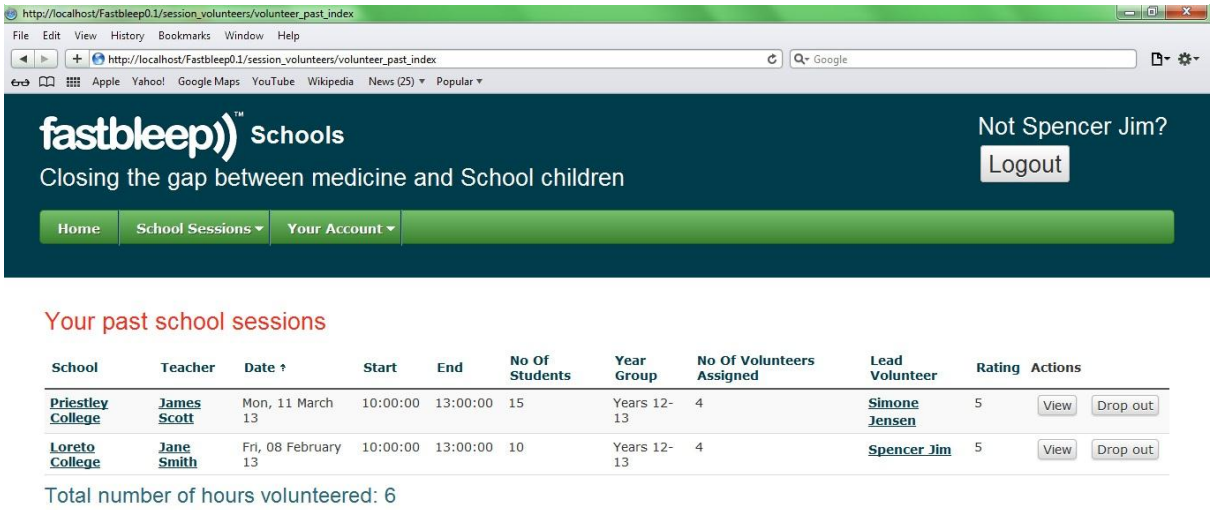
User Logout

User Change Password Form



Appendix C: System Testing on Different Web Browsers

Safari

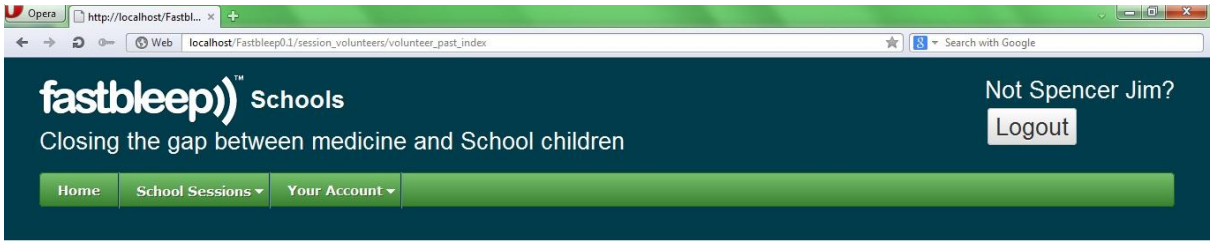


Your past school sessions

School	Teacher	Date	Start	End	No Of Students	Year Group	No Of Volunteers Assigned	Lead Volunteer	Rating	Actions
Priestley College	James Scott	Mon, 11 March 13	10:00:00	13:00:00	15	Years 12-13	4	Simone Jensen	5	View Drop out
Loreto College	Jane Smith	Fri, 08 February 13	10:00:00	13:00:00	10	Years 12-13	4	Spencer Jim	5	View Drop out

Total number of hours volunteered: 6

Opera

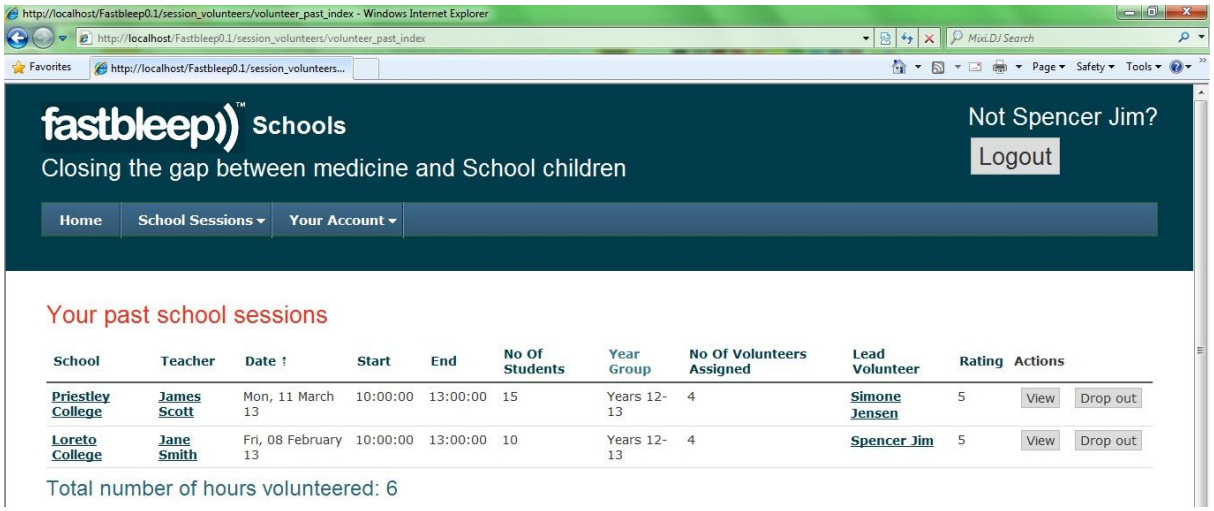


Your past school sessions

School	Teacher	Date	Start	End	No Of Students	Year Group	No Of Volunteers Assigned	Lead Volunteer	Rating	Actions
Priestley College	James Scott	Mon, 11 March 13	10:00:00	13:00:00	15	Years 12-13	4	Simone Jensen	5	View Drop out
Loreto College	Jane Smith	Fri, 08 February 13	10:00:00	13:00:00	10	Years 12-13	4	Spencer Jim	5	View Drop out

Total number of hours volunteered: 6

Internet Explorer

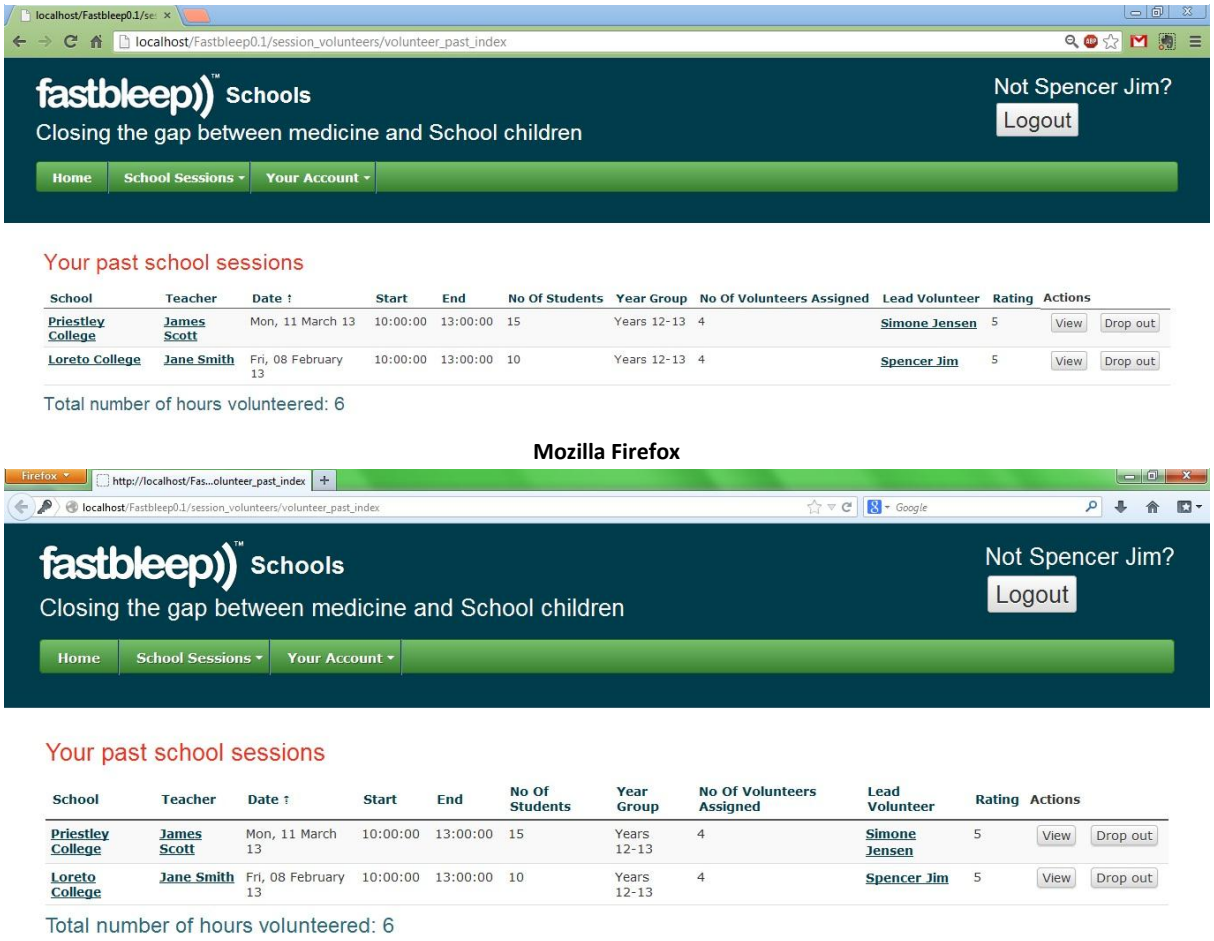


Your past school sessions

School	Teacher	Date	Start	End	No Of Students	Year Group	No Of Volunteers Assigned	Lead Volunteer	Rating	Actions
Priestley College	James Scott	Mon, 11 March 13	10:00:00	13:00:00	15	Years 12-13	4	Simone Jensen	5	View Drop out
Loreto College	Jane Smith	Fri, 08 February 13	10:00:00	13:00:00	10	Years 12-13	4	Spencer Jim	5	View Drop out

Total number of hours volunteered: 6

Google Chrome



Mozilla Firefox

Appendix D: AMEE Abstract

Technology IS the answer: A computer science intervention to support fairer access to medical education

(Oana Bradulet, Radu Asandei, Elspeth Hill, Suzanne Vaughan, Andy Brass)

Background

A recent report into UK Higher Education and social mobility calls upon medical schools to engage with school children, opening the profession to those traditionally excluded. However, decreasing budgets and larger student:teacher ratios mean activities to widen participation are low priority. Several successful initiatives have used volunteer medical students, however the significant administration time and effort required are a challenge to sustaining these programmes.

Summary of work

This collaboration between BSc computer science students and an organisation of medical students running a regional Widening Participation (WP) programme aimed to understand the requirements of the administrators, volunteers and school teachers in order to develop functional software that would meet these needs. Data were gathered through formal and informal feedback sessions, observations and documentary analysis of email and social networking posts.

Summary of results

Interdisciplinary working was extremely beneficial. The sustainability of the WP programme improved and administrator turnover reduced. Time and effort previously spent using multiple programmes and manually creating reports or emails could be spent engaging in the more rewarding activities in schools. Recruitment and retention of volunteers also increased. All stakeholders reported an improved experience.

Discussion

Technology is increasingly important in creating and sustaining activities across multiple communities.

Conclusions

Working in partnership with computer science is one way to develop more engaging and meaningful activity for multiple stakeholders.

Take-home messages

Collaborating across domains is a key strategy that medical schools should consider if they are to maintain WP activities in a time of decreasing resources.