

**UNIVERSITY OF SURREY**

**Faculty of Engineering and Physical Sciences**

**Department of Computing**

BSc Undergraduate Programmes in Computing

Module COM3001: 45 credits

**COM3001: Professional Project**

2014-15

FHEQ Level 6 (Year 3)

Student: **Chara Katiri**

Email: [ck00113@surrey.ac.uk](mailto:ck00113@surrey.ac.uk)

URN: 6166668

**Document Control**

| Version History | | | |
| --- | --- | --- | --- |
| Version | Date | Reason for issue / Change Request | Issued By |
| 0.1 | 29.10.2014 | Document setup & contents creation. | CK |
| 0.2 | 30.10.2014 - 10.11.2014 | Requirements gathering. | CK |
| 0.3 | 13.02.2015 | Draft table of contents. | CK |
| 0.4 | 16.2.2015 -21.02.2015 | Update of table of contents and creation of introduction, literature review and system analysis. | CK |
| 0.5 | 22.02.2015 | Creation of system design. | CK |
| 1.0 | 23.02.2015 | Update of diagrams. | CK |
| 1.1 | 19.03.2015 | Update of content in literature review. | CK |
| 1.2 | 20.03.2015 | Update of requirements table. | CK |
| 1.3 | 21.03.2015 | System design update. | CK |
| 1.4 | 21.03.2015 | Completion of ‘connection to the database behind the firewall’ | CK |
| 2.0 | 10.04.2015 | Update based on feedback from Draft report submission (Chapter 1-3) | CK |
| 2.1 | 11.04.2015 | Update based on feedback from Draft report submission (Chapter 3-4) | CK |
| 2.2 | 11.04.2015 | Update based on feedback from Draft report submission (Chapter 4-6) | CK |
| 2.3 | 12.04.2015 | Functional and non-functional requirements update. | CK |
| 2.4 | 02.05.2015 | Build tools and libraries | CK |
| 2.5 | 05.05.2015 | Update on References | CK |
| 2.6 | 09.05.2015 | Update on Database creation and implementation | CK |
|  |  |  |  |

Version History

| Document Reviewers | | | | |
| --- | --- | --- | --- | --- |
| Action | Name | Role | Date Received | Comments |
| Reviewer | Dr Steve Wesemeyer | Supervisor | 23.03.2015 | See CK-Formative-Feedback.pdf |
|  |  |  |  |  |
|  |  |  |  |  |

| Document Approval | | | | |
| --- | --- | --- | --- | --- |
| Action | Name | Role | Signature | Date |
| Approver | Dr Steve Wesemeyer | Supervisor |  | 23.03.2015 |
| Approver | Dr Anna Lisa Ferrara | Examiner |  | 25.05.2015 |
| Approver | Non known | Examiner 2 |  | 25.05.2015 |

|  |  |
| --- | --- |
| Project Owners | |
| Role | Name |
| Student | Chara Katiri |
| Supervisor | Dr Steve Wesemeyer |

|  |  |  |
| --- | --- | --- |
| Contact Details | | |
| All queries in relation to this document should be addressed in the first instance to: | Name | Chara Katiri |
| Department | Computing, University of Surrey |
| e-mail | ck00113@surrey.ac.uk |
| Address | University of Surrey, Guildford, Surrey, GU2 7XH |

|  |  |  |  |
| --- | --- | --- | --- |
| Document Name: COM3001 Professional Project. Project Requirements Document | Security Level: Private | Version Date: 12/04/2015 | Version: 1.8 |

**If printed this is an uncontrolled document**

##### **Acknowledgements**

##### **Table of contents**

1. Introduction 9

1.1. Motivation 9

1.2. Project idea and description 9

1.3. Project objectives and goal statement 11

1.4. Project benefits 11

1.5. Project stages 11

2. Literature review 14

2.1. Background research 14

2.2. Tackling the problem 19

2.3. Acceptance and usability concerns 19

2.4. Research on technology 20

2.4.1. Types of MVC Frameworks 20

2.5. Spring MVC Framework 22

2.5.1. Benefits of Spring MVC framework 24

2.5.2. Features of Spring MVC framework 25

3. System analysis 28

3.1. Current process 28

3.2. System requirements and analysis 28

3.2.1. Functional and non-functional requirements 29

3.2.2. Feasibility analysis 31

3.3. Summary of deliverables 32

3.4. Stakeholders 32

4. System design 34

4.1. Design method 34

4.2. Design plan 34

4.3. Diagrams 35

4.3.1. Class diagram 35

4.3.2. Use case diagram 36

4.3.3. Architecture and programming language 37

4.4. System risks and issues 39

4.4.1. System risks 39

4.4.2. System issues 39

4.5. System risks, dependencies and timescales 39

4.5.1. System risks 39

4.5.2. System constraints 40

4.5.3. System dependencies 40

4.5.4. System timescales 40

5. System implementation 42

5.1. Implementation plan 42

5.2. Build tools and libraries 42

5.3. Database creation and integration 44

5.1.1. Application Security 45

5.4. Success criteria 45

6. Test and evaluation 47

6.1. Requirements testing 47

6.2. Security testing 52

6.3. UAT testing 52

7. Conclusion 53

7.1. Achievements, review of objectives 53

7.2. Issues and shortcomings 53

7.3. Future work and suggestions for improvement 53

7.4. Final evaluation and conclusions 53

7.5. Summary 53

8. Bibliography 55

9. Appendices 60

9.1. Response from the IT Services. 60

9.2. Creation and remote connection to the database behind the firewall 61

9.3. Project Plan 63

##### **Table of figures**

(CK: Revisit this)

Figure 1: Overview of the Spring Framework **Error! Bookmark not defined.**

Figure 2: Request processing workflow in Spring MVC 27

Figure 3: Context hierarchy in Spring Web MVC **Error! Bookmark not defined.**

##### **Acronyms**

(CK: make sure they’re in alphabetical order)

|  |  |
| --- | --- |
| Acronym | Description |
| AD | Active Directory |
| AOP | Aspect Oriented Programming |
| Apache OFBiz | The Apache Open For Business Project |
| BRD | Business Requirement Document |
| CSS3 | Cascading Style Sheets 3 |
| DAOs | Data Access Objects |
| DI | Dependency Injection |
| ERP | Enterprise Resource Planning |
| HLD | High Level Design |
| HTTPS | Hyper Text Protocol Secure |
| IETF | Internet Engineering Task Force |
| IoC | Inversion of Control |
| IST | Integrated Systems Testing |
| JDBC | Java DataBase Connectivity |
| JMS | Java Message Service |
| JSF | Java Server Faces |
| JSR | Java Specification Request |
| LLD | Low Level Design |
| MoSCow | Must, Should, Could or Won’t |
| MVC | Model View Controller |
| MySQL | My Sequel |
| MySQLi | My Sequel Improved |
| OOP | Object Oriented Programming |
| ORM | Object Relational Mapping |
| OXM | Object/XML |
| POJO | Plain Old Java Object |
| R&D | Research & Development |
| RAD | Rapid Application Development |
| RDBMS | Relational Database Management System |
| SDLC | Systems Development Life-Cycle |
| UAT | User Acceptance Testing |
| UBE | Unsolicited Bulk Email |
| JEE | Java platform Enterprise Edition |
| WAP | Web Application Framework |

# Introduction

The purpose of this document is the detailed description of the project’s plan to design, develop and deliver a digital Pinboard website as a method to minimize the number of junk emails sent and received between students in the University of Surrey. The project is developed as part of ‘COM3001 Professional Project’ module, taught by the University of Surrey, Department of Computing.

## Motivation

Nowadays, electronic mail is the predominant digital communication platform. According to Reuter’s survey [[1](" \l "Reu15)], 85% of human population is connected online and communicate through email despite the number of other social channels available. The Internet is overflowing with 100 million emails send and received daily [[2](" \l "The151)], of which 97.4 millions are spam/junk [[3](" \l "Ese15)].

The question investigated is how to reduce the number of ‘junk’ emails send and received within the University of Surrey environment.

## Project idea and description

The lack of an organised way to advertise second hand books or requests for house sharing, leads the students to the creation and distribution of unsolicited bulk email (UBE) emails. These emails are categorised as junk emails.

In this project, a digital Pinboard website is developed as a method to minimize the number of ‘junk emails’ send and received daily within the University of Surrey. Junk emails have a negative impact on the size of users’ mailbox and the management of incoming emails. The Pinboard website, in the form of a marketplace, attempts to resolve this issue and allows students in the University community to connect, buy and sell second hand books and advertise room swaps on campus.

The model canvas shown below as a visual chart (Figure 1), describes the main elements of the Pinboard website idea. The creation of the template in the early stages of the project helped identify potential trade-offs [[4](" \l "Ost)] and led to the refinement of the requirements’ list to enable the delivery of the most important priorities.

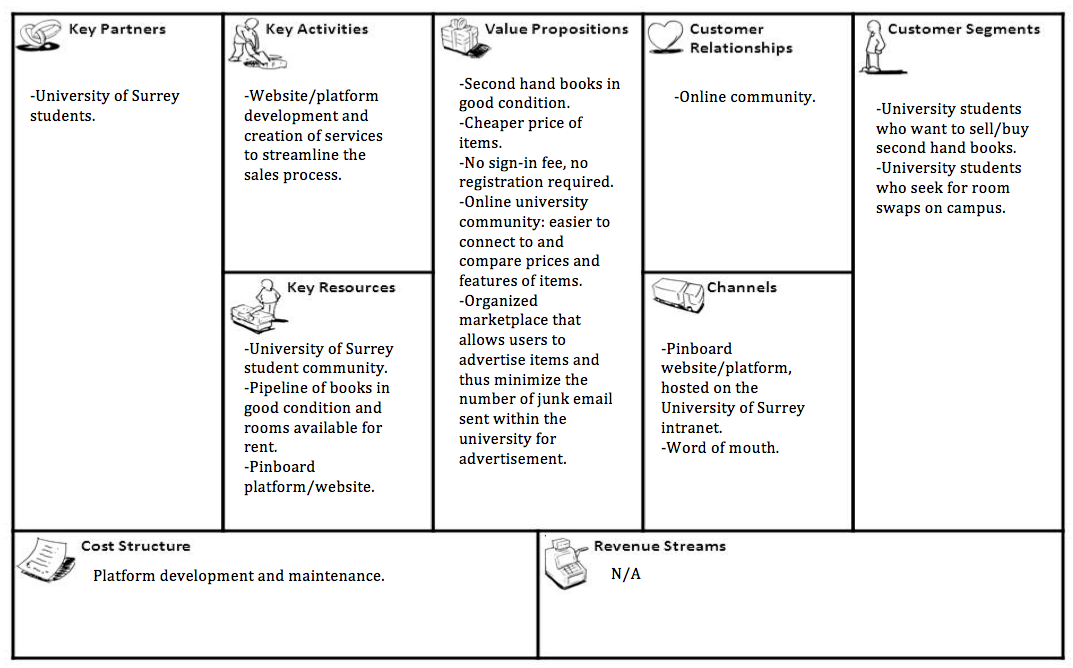


Figure 1: Pinboard - model canvas

## Project objectives and goal statement

The objective of the project is the creation of a dynamic website as a University-specific marketplace. The digital Pinboard website offers an organised way for students to sell second hand books. It also provides functionalities that allow students to advertise available rooms and make house hunting more efficient.

The use of the services offered by the website have the potential to significantly reduce the number of ‘junk’ emails sent daily between students regarding sales of second-hand books and requests for room swaps on campus.

The personal development objective is the successful completion of the full development lifecycle from general project management and solution design, to the development and delivery of the website. Another personal objective is the use of new technologies as a challenge to perform beyond the comfort zone and gain new skills. The challenge, is the development of the website using Aspect Oriented Programming (AOP), as opposed to Object Oriented Programming (OOB) that was extensively taught and practiced during Year 1 & 2. AOP as well as Spring MVC Framework were introduced only during Year 3,Sem1, and they were used for the development of this project. The use of technologies to achieve scalability of the website regardless the screen size (mobile phone/tablet/desktop) or Operational System (iOS/Android/Windows) is another personal objective. CSS3 is used to enable the website adapt to different screen sizes and resolutions whether rendered on mobile phone devices, tablets or desktops.

## Project benefits

The project benefit is the introduction of a new communication channel that allows students to sell second-hand books and search for housemates/swap rooms on campus. Therefore, the use of the website can help minimize the creation of ‘junk emails’ sent to advertise second-hand books and room swaps. The deployment of the website on the University’s intranet and the services it provides can also promote the replacement of the physical Pin-boards on campus and thus reduce paper copies and promote environmental friendly behaviour. Lastly, the website can help reduce the network traffic and disk space used by ‘junk’ emails and therefore minimize the cost of storage required as well as other management activities carried out by the IT Services team.

## Project stages

The management of this project is based on the Systems Development Life-Cycle (SDLC) [[5](" \l "Sea15)]. The diagram below shows the SDLC stages. The first step was the investigation of an existing problem that can be improved.

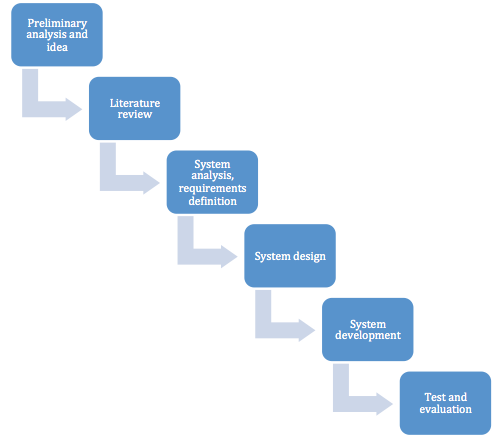


Figure 2: Project Stages

1. *Preliminary analysis and idea*: To help minimise the creation of junk emails for second hand books and room swaps, the development of the Pinboard website was suggested.
2. *Literature review:* Background research helped identify websites, which offer similar services and recognise their strengths. Examples of existing open-source websites are Apache OFBiz, osCommerce and Magendo. Other e-commerce websites examples are Gumtree, Preloved and eBay.
3. *System analysis, requirements definition:* Functional and Non-functional requirements were defined and analysed to create a Business Requirement Document (BRD).
4. *System design:* To support the BRD, features of the website were defined and screen layouts as well as process diagrams are created.
5. *System Development:* In preparation for the development of the website, a request was made to the IT Services team to enable access to a Tomcat server and a MySQL database backend. Following the creation of Pinboard’s home page, the login functionality for the user was developed. Appropriate measures were taken to ensure that input text submitted by users is secure and the website is protected from attacks. A local as well as a remote database instance was created and additional functionalities were added to improve the usability of the website. These functionalities include creation of bookmarks for the available items, the option to allow the users change their password (CK: double check reqs) and additional buttons to share the website on social media channels.
6. *Test and evaluation:* Following the completion of development the system was tested to ensure that each requirement is met. Test of functional requirements ensure that different web browsers can be used to open the website and unexpected behaviour or crash of the web client is avoided. The full list of functional requirements can be found in section *‘3.2.1 Functional and non-functional requirements’*. Tests were also undertaken to tackle security concerns such as unauthorized access to the website or SQL injection by attackers. Finally, User Acceptance Testing (UAT) was performed to ensure that early adopters are happy with the service and functionalities offered by the website.

# Literature review

## Background research

Background research was undertaken to identify other solutions that offer similar functionalities and understand how those websites are developed. Identification of their strengths and weakness helped identify the must-have features of the Pinboard website. The main areas investigated were the target audience and key functionalities that the users would like to see on the website. The comparison made was between a number of open source e-commence solutions such as Apache OFBiz, osCommerce, Magendo [[6](" \l "Wik15)] and other e-commerce websites such as Gumtree, Preloved and eBay. The strengths and weakness of the services are shown below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Functionalities** | **Apache OFBiz** [[7](" \l "Ofb15)] | **osCommerce** [[8](" \l "Osc15)] | **Magendo** [[9](" \l "Mag15)] | **Pinboard** |
| First Release date | 2007 | 2000 | 2007 | - |
| Licence | Apache License 2.0 | GNU GPL | Open Software Licence | - |
| Language | Java | PHP | PHP | Java, CSS3, HTML5Java, JavaScript, HTML, CSS |
| Web Application Framework (WAP) | Appache OFBiz | - | Zend Framework | Spring MVCSpring MVC Framework |
| Free database backend support | Apache Derby, MySQL, PostgreSQL | MySQL | MySQL | MySQLMySQL, Apache Tomcat |
| Non-free database backend support | MySQL, Oracle | - | MySQL, MySQLi | -- |
| Out-of-the-box shopping cart | Yes | - | Yes | - |

Comparison between open source e-commerce solutions:

Comparison between e-commerce websites:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Functionalities** | **Gumtree** [[10](" \l "Wik151)] | | **Prevolved** [[11](" \l "Sec15)] | | **Ebay** [[12](" \l "Wik152)] | | **Pinboard** |
| Type of site | Classifieds | | Classifieds | | E-commerce | | Classifieds |
| Registration | Optional | | Required [[13](" \l "Sec151)]. | | Required to sell. Guest checkout avaialble. | | Not required. Student’s University email address will be used to login. |
| Target audience | - Initial target audience: Australians, New Zealanders, South Africans.  - Current target audience: Business customers and general public.  - Current target users in the UK, Poland, France, Canada, Australia, New Zealand, South Africa, Hong Kong, Singapore. | | Business sellers and general public. | | Business sellers and general public | | University of Surrey students and staff. |
| Number of users | 13.7 million [[14](" \l "Lon15)] | | 5.5 million | | 155.2 million [[15](" \l "Sta15)] | | University of Surrey students and staff. |
| Categories | -Goods for sale;  -motors listings;  -jobs advertisements;  -property advertisement. | | Over 500. [[16](" \l "Pre15)]  Initially for second hand or vintage items.  Is currently one of the most popular websites used to advertise pets, horses, and livestock. | | Over 35+.  Goods for sale, (new/used) auction, “Buy it now” shopping(fixed price), ticket trading, money transfers etc. | | Second hand books;  Rooms available for rent. |
| Featured adverts | -Free use of the website.  Featured adverts:  -advert in the top of search page;  -advert on the homepage;  -urgent advert;  -site search ranking boost. | | -Free use of the website.  Featured adverts:  -Option available for a yearly membership upgrade (buyers £5, sellers £25). Once an advert is created is only available to membership holders. After 10 days the advert is available to non-membership users too. | | -Free use of the website.  Featured adverts:  -When the seller adds new items and when they’re sold he is charged by an invoice fee. | | - Free use of the website. |
| Delivery services | | Collection from the seller only. | | Shipping | | Shipping | Collection from the seller only.  Students are located on campus and deliveries are made in person (meeting between student seller and student buyer). |
| Payment method | Online payment. | | Online payment. | | Online payment. | | Students are located on campus and payments are made in person. |
| Presence in social media | Facebook,  Twitter,  Google+,  Pinterest. | | Facebook,  Twitter,  Google+,  Instagram. | | Twitter,  Google+,  Intagram,  LinkedIn. | |  |

## Tackling the problem

The development of Pinboard website provides an alternative process to email distribution for students that wish to sell their second-hand books or request room swaps on campus. By providing an organised system, the target audience can navigate to Pinboard website and check numerous options before they make their decision on which book to purchase / room to swap. The organised service helps minimize the number of bulk emails created by student-sellers that wished to advertise books or room swaps.

## Acceptance and usability concerns

The key concern is for the IT Services/Systems Team to accept the deployment of the website on Surrey’s intranet. If it’s accepted, the arising concern is the level of usability of the website by the end users. However, if the deployment of the website is not approved, the project will be seen as an opportunity for hands-on experience, management and completion of the development lifecycle for a website. (CK: However…website: review this)

## Research on technology

## Types of MVC Frameworks

There are 2 types of MVC Frameworks (i) UI Component oriented MVC and (ii) Action oriented MVC [[17](" \l "Ora15)]. The differences between Component Oriented Frameworks and Action Oriented Frameworks are shown below.

The table is taken from the ‘MVC 1.0/JSR 371’ presented during JavaLand 2015 by David Database [[18](" \l "Sli15)]. The same table used during Devonxx 14, by Mafred Riem [[19](" \l "DMV15)].

|  |  |  |
| --- | --- | --- |
|  | **UI Component Oriented MVC (JSF)** | **Action (or Request) Oriented MVC (JSR)** |
| JSF, JSR examples [36] | -JavaServer Faces JSF  -Wicket  -Tapestry | -Spring MVC.  -Apache Struts (known as Jakarata Struts).  -Stripes. |
| JSF, JSR characteristics [[20](" \l "Doc15)], [[17](" \l "Ora15)], [[18](" \l "Sli15)] | -Automatic request parameter procession.  -View kept around.  -Component libraries that implement re-usable behaviour.  -Components render HTML/JavaScript.  -Automatic input conversion.  -Automatic input validation.  -Page centric. | - Manual request parameter processing.  -No view kept around.  -Limited support for re-usable behaviour.  -Developer responsible for all HTML / JavaScript.  -No automatic input conversion.  -No automatic input validation.  -Request centric. |

The main difference between the two MVC Frameworks is that in Component Oriented MVC (JSF) the framework provides the Controller by default and therefore the developer has very little control over the output of the objects. The diagram below shows the process in UI Component Oriented MVC as Exposed in JSF.

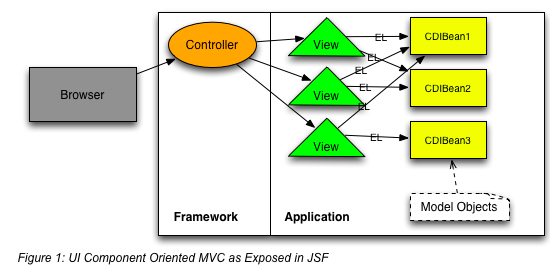


Figure 3: UI Component Oriented MVC as exposed in JSF

On the other hand, Action Oriented MVC (JSR) is configuration-intensive but enables the developer to define the controller(s), which allows full and detailed control of the output and help him focus on business logic. The diagram below shows the process in Action Oriented MVC.

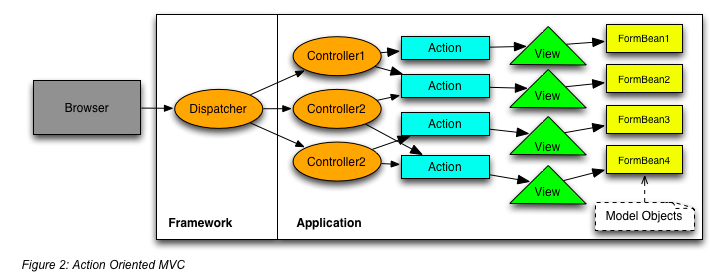


Figure 4: Action oriented MVC

## Spring MVC Framework

The client and web server website is developed using Java and the Spring MVC Framework.

Spring MVC is a lightweight action oriented framework (also known as request oriented implementation and JSR). *Spring Framework* is an open source, Aspect Oriented Programming (AOP) framework for Java based applications. *Spring MVC Framework* is an extensible MVC framework and a component of Spring that is used for the creation of web applications. It’s developed on core Spring functionality that provides technologies for views (JSP) [[21](" \l "Spr14)].



Figure 5: Overview of Spring MVC Framework

The Spring Framework consists of a number of modules: Data access/integration; Web; AOP; Aspects; Instrumentation, Messaging; the Core container and finally the Test module.

* In Data access/integration module provides support to: (i) JDBC, (ii) ORM, (iii) OXM, (iv) JMS and (v) Transaction module.

1. The Java DataBase Connectivity technology (JDBC) defines how a client accesses the database and provides methods to query and update the database.
2. The Object Relational Mapping (ORM) converts data between incompatible types and systems. In particular, this technique is beneficial in modern systems where systems include a number of subsystems.
3. The Object/XML mapping (OXM) converts objects to XML and vice-versa.
4. The Java Message Service (JMS) sends messages between two or more clients and enables reliable connection between different components.
5. The Transaction module coordinates the transactions for management APIs and Java objects.

* Aspect Oriented Programming (AOP) manages the application challenges, such as security, logging and management, into aspects and helps create modules that enable reusability. AOP complements OOP (Object Oriented Programming) by creating a new program structure. The difference between OOP and AOP is that in OOP the key element (unit of  modularity) is the class however in AOP the key element is the aspect.  AOP framework is an important component in Spring and completes the Inversion of Control (IoC) to provide a capable middleware solution [[22](" \l "Spr151)].
* The ‘Core Container’ is the key module in Spring Framework and consists of the following sub-modules: Beans, Core, Context, SpEL. BeanFactory, as the most important component of core container makes use of ‘spring-beans’ and ‘spring-core’ sub-modules to provide a number of essential functionalities and features for Inversion of Control (IoC) and Dependency Injection (DI). ApplicationContext is equally important and extends the BeansFactory. The ApplicationContext provides internationalisation for text messages; enables access to resources like URLs and files; and supports event publication to the registered listeners. [[23](" \l "Spr15)]
* Test module: the purpose of the test module is to support the services for integration and unit testing.

## Benefits of Spring MVC framework

Spring MVC Framework was chosen over other MVC technologies, such as Apache Struts, due to the support features it offers. Apache Struts is a sophisticated framework, strictly focused on presentation. On the other hand, Spring is a “*Lightweight inversion of control and Aspect Oriented Container Framework*” [[24](" \l "Wit15)]. It’s a popular framework used by a lot of web developers because it offers quality from design to implementation, promotes best practices and it’s adaptable. A detailed description of other benefits that Spring MVC Framework offers is shown below:

* *Separation of roles:* The use POJO (Plain Old Java Object) and the creation of specialised objects help escape duplication and result in clear separation of roles, from presentation to business logic. Those objects are implemented in specialized classes, such as Model View Controller (MVC), DispatcherServlet and Handler Mapping. In addition to this, the AOP module allows definition of ‘pointcuts’ and method-interceptors to achieve separation of code when functionalities are implemented [[25](" \l "Doc151)].
* *Modularity:* Spring Framework offers a number of modules. This is beneficial for the developers because they have the option to choose only the modules relevant to their application rather than having to include the entire framework.
* *Reusable code:* The use of existing business objects allows reusability of code and therefore reduces definition of those objects and duplication of code. The application classes are configured as JavaBeans and therefore reference can be made from the web controller to the business object.
* *Flexibility and maintainability:* Name and value maps support integration with any View and therefore the system flexibility is increased. Also, implementation of a number of separate controllers instead of a single controller for everything helps improve maintainability of the application [[21](" \l "Spr14)].

## Features of Spring MVC framework

* *Model, View, Controller (MVC) design pattern:*

Model View Controller is a well-established software architectural pattern. It’s used in web application development in order to separate the business logic from the user interface [[26](" \l "Cop15)]. The MVC consists of three modules: Model, View and Controller. The Controller is the front servlet and acts as an interface between the View and Model.

The diagram below shows the integration between the modules.

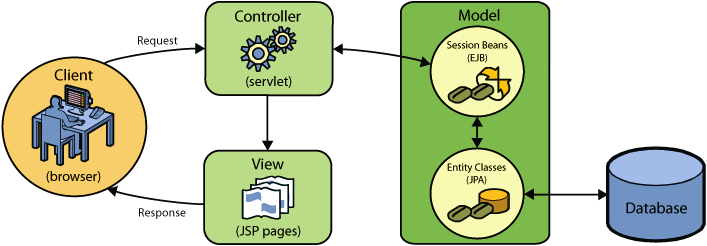


Figure 6: Context hierarchy in Spring Web MVC

The client sends requests to the Controller. The Controller then forwards the input to JSP pages. In Spring Web MVC terminology, JSP pages are named Views and are managed by the Controller. In response, the View (JSP) calls the Model.

The Model (JavaBean) represents the state (data) and business logic of the application. The Model connects to the database to retrieves/saves data.

The View (JSP) generates the response and sends it back to the client.

In complex applications, the Controller also manages the Service entity whenever it needs to perform business logic. This helps improve maintainability, as the business logic is not contained directly in the controller [[27](" \l "DVr15)].

The diagram below shows the detailed list of actions that each is module is responsible for.

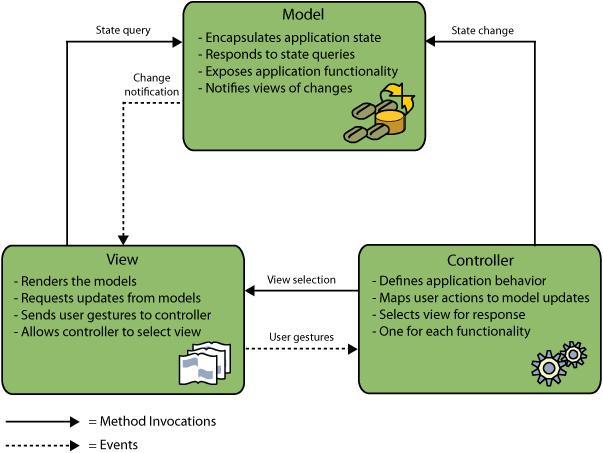


Figure 7: Description of modules

* *DispatcherServlet (How the Controller in Spring MVC Framework facilitates the Controller in MVC):* (CK: “Ensure to explain how controllers in Spring are related to Controllers in MVC”. Ck: double check if this is answered here. )

How the Controllers in Spring MVC Framework are related to the Controllers in MVC: The Controller(s) in Spring MVC Framework make use of the Front Controller to dispatch requests to Controllers in MVC [10]. This process is managed by the DispatcherServlet, which and acts as the front controller and is configured in XML. The Front controller receives the incoming request from the client and then delegates the request to a Controller. The Controller receives and handles the request, performs business logic and creates a new model. In order to delegate rendering of response, the Controller forwards the model to the Front Controller. The Front Controller selects a View and forwards the model in order to create a render response. Finally the View returns control to the Front Controller and the response is sent to the client.



Figure 8: Request processing workflow in Spring MVC

# System analysis

## Current process

Currently, there’s no organised process for the University of Surrey community to advertise second hand books or request room swaps on campus. As a result, the sellers/seekers create and distribute bulk emails (known as UBE) hoping that someone will get back to them. This process generates numerous unwanted emails for students that are not interested, is possible that it overloads the network as well as the storage servers.

## System requirements and analysis

The requirements listed below were used as the basis for the development and testing of the web application, to determine the completion of the project and assess its success.

* *Requirement Type:* Functional or Non-Functional.
  + Functional: Requirements are classified as Functional if their aim is to provide information on what the system should do [[28](" \l "Leg15)]. Examples include: authentication and authorization processes, historical data, audit tracking, database requirements etc. [[29](" \l "Sqa15)].
  + Non-Functional: Requirements are classified as non-functional if they cover constraints of the solution, targets and ways to control its mechanisms. Examples include performance, scalability, maintainability etc. [[30](" \l "ReQ15)].
* *Shall/Must terminology*: These keywords were standardised by the Internet Engineering Task Force (IETF) to specify the level of requirements. Requirements marked as ‘Shall’ indicate absolute requirement. ‘Must’ indicates desired requirement [[31](" \l "Iet15)].
* *MoSCoW:* ‘Must or Should Could or Won’t’ is a requirement and prioritization method. The prioritization shows the importance of each requirement however it does not mean that requirements classified as ‘Must’ will be developed first; it means that by the completion of the project they must be delivered. Requirements classified, as ‘Won’t’ should be listed even if they won’t be met during this sprint. Those requirements will be used as suggestions that ‘Would’ be implemented as part of future development to improve the system [[32](" \l "Fam13)].

## Functional and non-functional requirements

(ck: revisit the Shall and Must column

(MoSCow: M) = (Shall in S/M terminology) ??

(MoSCow: S) = (Must in S/M terminology)??)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Req. no.** | **Requirements Description** | **Req. Type** | **Shall/Must** | **MoSCoW** |
|  | **Database** | | | |
| 1 | The centralized database that contains the products must update automatically and in real time. | Non-functional | Shall | M |
| 2 | The system must update the list of products so that a product becomes unavailable after a user purchases it. | Functional |  | M |
|  | **System permissions and roles** |  |  |  |
| 3 | The system must be available only to the University of Surrey students and staff. | Functional |  | M |
| 3.1 | The system must offer 2 different permission roles. One for admin-role and one for standard user-role. | Functional |  | M |
| 3.2 | The system must allow admins full control over the website, including deletion of user accounts and items. | Functional |  | M |
| 4 | The system must forbid standard-users from accessing admin pages. | Functional |  | M |
|  | **Login - Logout** |  |  |  |
| 5 | The system must forbid access to users with invalid username. | Functional |  | M |
| 6 | The system must provide a ‘Forgot my password’ functionality to allow users reset their password. | Functional |  | M |
| 7 | The system must provide logout functionality. | Functional |  | M |
| 8 | The system won’t require the users to register. The login process will be based on the existing UoS Active Directory (AD) groups. New students and staff will be automatically added in the AD group and they will be able to access the website. | Functional |  | W |
|  | Validation |  |  |  |
| 9 | The system must validate the user information upon login. | Functional |  | M |
| 9.1 | A UoS Username must be provided in order to login. | Functional |  | M |
| 9.2 | Password must be provided in order to login. | Functional |  | M |
| 9.3 | Password must be at least 8 characters long. | Functional |  | M |
| 9.4 | Password must be hashed in the database. | Functional |  | M |
| 9.5 | The system must check if the username is valid by checking the list of usernames in the UoS AD group. | Functional |  | M |
| 9.6 | The system must decline usernames that are not valid. | Functional |  | M |
|  | Users |  |  |  |
| 10 | The system could provide a ‘My profile’ area for each user. The personal information of the user will be shown on that page (Name, Contact details, email address, telephone number). (CK: revisit this) | Functional |  | C |
|  | Items |  |  |  |
| 11 | The system must provide information about the product: Title, Description, Price, Category and Seller contact information. | Functional |  | M |
| 12 | The system must ask the seller to fill specific fields regarding the product: Title, Description, Price, Category and Seller contact information such as email address and/or mobile phone number. | Functional |  | M |
|  | Sort & Search items |  |  |  |
| 13 | The system must allow the users to search products based on categories e.g. books or room swaps. | Functional |  | M |
| 14 | The system must provide a search functionality (text-box search) to allow the users enter keywords. | Functional |  | M |
| 15 | The system could enable the users to sort the products based on price (Low to High, or High to Low). | Functional |  | C |
|  | Favourites |  |  |  |
| 12 | The system should allow users to bookmark items as ‘Favourites’. | Functional |  | M |
| 13 | The list of Favourites must be private and not visible to others apart from the user who created the personal Favourite. | Non-functional |  | M |
|  | **Terms & Conditions** | | | |
| 3 | The system must ask the users (byers & sellers) to agree with ‘The Sales of Goods Act 1979’ and adhere to it: “*"traders must sell goods that are as described and of satisfactory quality”* [[28](" \l "Leg15)], [[33](" \l "Gla15)]. The Terms & conditions must be accessible, meaningful and fair. (CK: Delete this?) | Non-functional |  | M |

## Feasibility analysis

Feasibility study [[34](" \l "Ext15)] is conducted as part of the assessment process. The aim of the project analysis was to identify trade-offs and outline alternative options. The feasibility analysis helped recognise issues and potential risks in the early stages of the software development system. One of the first risks identified was that the Service Team might not accept the deployment of the website to the University of Surrey intranet. After the discussion of the project with the Service Team, the feasibility of the project was also discussed with the project supervisor and the decision made was to go ahead with the development of the digital Pinboard.

Even though the Pinboard website might not be deployed on the University’s intranet, it is an opportunity for the final year computing student to plan and deliver the project through the different phases SDLC of systems development lifecycle. It was also an opportunity to achieve a challenge and develop a project based on a new Framework without previous Spring MVC experience. (CK review this)

## Summary of deliverables

|  |  |
| --- | --- |
| **Deliverable name** | **Description** |
| Project Report | The project report provides detailed description of the project including literature review; objectives; system analysis, design and implementation; test and evaluation; reflections and achievements. |
| Project Design | The structure of the solution and the plan to implement the functional and non-functional requirements in order to meet the project objectives. |
| Project Video | The project video presents the key functionalities offered by the system created. |
| Definition of functionalities and services (CK: revisit this, might not needed) | The services delivered by the project should be clearly defined for the benefit of:  i) The customer: to describe how navigate thought and use the website  ii) The IT team: that will be responsible for the maintenance of the website if the deployment of the solution is accepted. |
| Specification of Roles and responsibilities (CK: revisit this, might not needed) | Clear specification of roles and responsibilities of:  i) The user and the policies that must adhere  ii) The IT Team that will be responsible for the maintenance of the website if the deployment of the solution is accepted. |

## Stakeholders

Stakeholders [[35](" \l "Dic15)]are individuals or organizations with interest in a project. Typically they are internal or external investors, employees, customers and suppliers. Stakeholders of the digital Pinboard solution include the target audience, which is a subset of students; sellers of second-hand books and those who wish to swap rooms on campus.

|  |  |
| --- | --- |
| **Stakeholder** | **Support Role** |
| A subset of University of Surrey students and academics. | A group of volunteers that wish to test the application and are interested in the functionalities that the application offers. |
| University of Surrey IT and Service Desk. | The support team if the deployment of the solution is accepted. |
| Chara Katiri | Computer Science Student, the developer. |
| Dr Steve Wesemeyer | Professional project supervisor and coordinator. |

# System design

## Design method

Waterfall methodology was chosen over Agile as the development approach. In the project environment, Waterfall methodology can be easily understood not only by the developer but also the client due to the clarity of the linear sequence.

In contrast to Agile methodology, in Waterfall, the complete scope of requirements is known in advance and therefore each phase must be fully developed and delivered before the developer moves to the next one. Tasks are divided chronologically in achievable chunks and delivered in a number of phases. Specific deliverables and review process makes each phase easy to manage [[36](" \l "stq15)].

The diagram below shows the Waterfall Software Development Life Cycle (SDLC) and the tasks required in each phase:

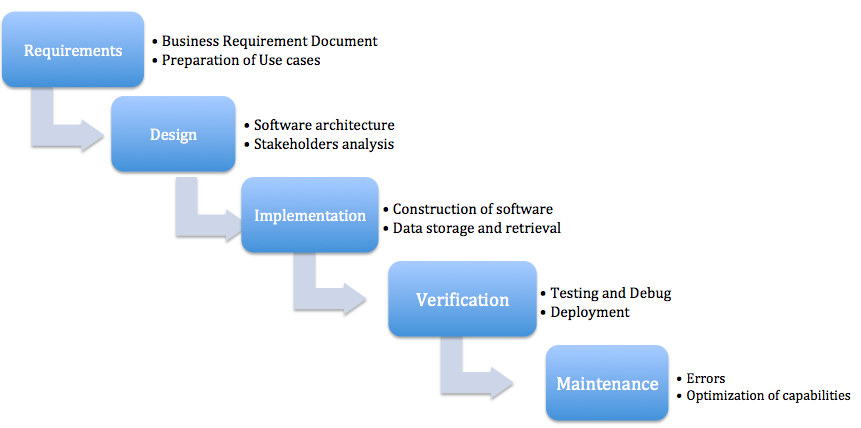


Figure 9: Waterfall Software Development Life Cycle

## Design plan

The purpose of the design phase was to define i) the Architectural design, ii) the High-Level design and iii) the Low-Level design [[37](" \l "Cso15)].

The Architectural design phase aimed to answer the most fundamental questions that affect the system as a whole. Some of the decisions made in this phase included: use of a database instead of regular files to store the content that the system will make use/generate; use of a computer instead of a distributed system of numerous computers to run the system.

In High-Level design (HLD) phase the system was divided into a number of modules to enable delivery of tasks in achievable chucks and therefore make the system more manageable.

Finally, in Low-Level design (LLD) phase the data structure, logic and algorithms for each module were defined for each module and class diagrams were created.

The main resource that the development plan was based on is the final year student - developer (CK) and the supervisor (SW) that supported and guided the project.

## Diagrams

## Class diagram

A class diagram was created as a modeling technique to help visualize the relationship between different types of objects and classes.

The Pinboard class diagram is shown below.

Relationship between classes: (to be completed)

## Use case diagram

Main actors of the system: Admin and standard users (sellers and buyers) are the key stakeholders of they system and they interact with each other.

* Admin: the admin has full control over the system. He is responsible for maintenance, he provides user support when required and can delete users and adverts if required.
* Standard users can be buyers or sellers. Seller: the seller makes use of the system to post advertisements and supply information about the items available for sale. When the item is sold the seller deletes the advert for that item.
* Buyer: the buyer uses the information supplied by the seller to decide which item to buy. The most useful features for the buyer are:

1. *Books and Rooms* categories that allow him to search items by category
2. *Search box* that allow him to search by keyword
3. *Favourites* that allow him to bookmark items as favorites.

These features help the buyers navigate faster to the item he is interested in buying. The buyer has no permissions to modify or delete any of the items listed for sale.

Pre-conditions:

* The user is a UoS student or academic (or admin) and therefore has a UoS username and email address.

Basic flow:

1. The user logs in using his UoS username.
2. The seller posts adverts such as second-hand books available for sale or room swaps requests.
3. The buyer searches the catalogue of available items.
4. The buyer can create bookmarks for items he is interested in and also view those bookmarks in a list.
5. The buyer decides which item to buy and gets in touch with the seller via email or a phone call.
6. The seller meets the buyer to deliver the item and collect the payment from the buyer.
7. The seller deletes the advert for the item sold as is no longer available.
8. If required, the admin has permissions to delete a user or an advert at any stage of this process.

The use-case diagram below shows the main functions of the system. (CK: to be updated using software specific for use case diagrams).

## System architecture

The typical Java-based Persistence Architecture consists of 5 layers: Application layer, DAO, ORM, Data Source and Database. (CK: The java-persistence architecture layers are now added. Can I keep the layers described above as well ?)

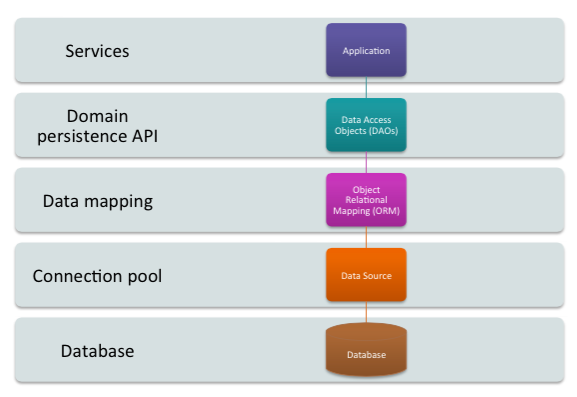


Figure 11: Java-based Persistence Architecture

* The Application layer consists of the core services that form the business logic of the application.
* The Data Access Objects (DAOs) consists of the database services and domain persistence API. The DAO is a specialised service used by the controller to access and retrieve data from the database.
* The Object Relational Mapping (ORM) is a powerful object-oriented functionality and its purpose is to convert data between incompatible types and systems. For example it can transform an object to XML, JSON or Java based on the needs of the system. It’s particularly useful in modern Web applications where different types of subsystems are part of the same system.
* The Data source layer is the connection pool and finally, the Database layer provides access to the data.

## System risks and issues

## System risks

* If the deployment of the website on the University of Surrey intranet is not approved, the deployment to early adopters cannot be completed and neither feedback can be collected.
* UAT depends on testers being available when required (CS students, volunteers).
* End user expectations unrealistic compared to solution.

*Contingency plan to mitigate risks:* The development of the website is based on Rapid Application Development (RAD) which allows visual prototypes to be shown to the stakeholders. This keeps stakeholders engaged and feedback can be received in early stages of the development lifecycle.

## System issues

*Collection of junk emails:* One of the initial aims of the system was to collect the number of the junk emails sent within the University. Then, the plan was to analyse the number of emails sent before and after the deployment of the website in order to measure the efficiency of the website through the functionalities it offers. A request was submitted to the IT services team to extract the average number of junk emails exchanged on Surrey’s network. In response the team explained that there is no way to pull out the junk emails as all Surrey accounts are automatically listed as ‘safe senders’ and they can’t be filtered as junk unless those emails are marked as junk by the receivers. (The email trail can be found in section 9.1. Appendix).

Mitigation ? (CK: mitigation to be reviewed)

## System risks, constrains, dependencies and timescales

## System risks

* Time constraints due to 5 other modules that run in parallel with COM3001. All the modules are assessed based on at least one coursework and an exam.
* Time scales for set up, development and deployment of the website are aggressive in order to meet the demands of COM3001 Professional Project deadlines and deliveries.

*Contingency plan to mitigate risks:*

* Timescales were set up with the work required by other modules in mind.
* If for any reason the timescales shift, the requirements categorised as ‘Could’ will not be implemented.

## System constraints

This section covers the most important system constraints:

* Spring Framework 4.0 is the latest version released and it requires minimum Java SE 6 and above. Spring Framework 4.0 supports fully Java 8 features and use of Java 7 or above is recommended [[38](" \l "Doc152)].
* Tomcat server is required. As shown in ‘The 2014 Leaderboard of Java Tools & Technologies’ [[39](" \l "Pag14)] Tomcat is the most popular application server used in Spring Framework. This is due to the easy of download and installation, tooling support and server configuration.
* MySQL server is required to save user data and items data into the database.

## System dependencies

A number of dependencies are required to configure authentication in Spring MVC and allow the application to function. Some of the dependencies are: [[40](" \l "Doc153)]

* ‘spring-security-confiq’ that is used for Spring Security namespace configuration.
* ‘spring-security-web’ that is required for applications that use Servlet API.
* ‘spring-jdbc’ that is used for Spring JDBC authentication,
* ‘DelegatingFilterProxy’ that is used as a filter to intercept the HTTP requests for authentication purposes

## System timescales

Following the gathering and definition of functional responsibilities a project plan is developed. Time limits for scheduling were created to ensure that the required time is allocated for the completion of each task. Milestones and deliverables were defined to keep track of the project progress and help get closer to the finished product. Then the Critical Path was defined. Activities included in it are critical and any delay in delivering those will have a negative impact on the rest of the project and will result delay in the delivery of the project [[41](" \l "May10)].

The aim of the project plan was to ensure balance between timescales and project quality. Details in the R&D phase were important but it was equally important to avoid destructions and remain focused on the project goal [[42](" \l "Tea15)].

The optimal strategy to estimate timescales for system development is: [[43](" \l "Bar09)]

Phase 1:

* + Research and planning: 5%
  + Solution design: 5%
  + Design: 25%

Phase 2: Frond-end and back-end development: 25%

Phase 3: Testing 30%

Phase 4: Contingency: 10%

The detailed project plan including Milestones and resources allocated for each task is shown chapter 9.3 Appendix.

# System implementation

## Development tools

This section covers a number of development platforms and build tools that were used for the development, formatting and management of the code

NetBeans is a software development platform. NetBeans was used for the development of Pinboard as a Java web application (with HTML, CSS and JavaScript). Applications in NetBeans are developed using a set of modular software components known as modules. NetBeans, supports the latest Java platform EE (JEE) 6 and 7 features, as required by Spring MVC, and provides excellent integration with Apache Tomcat web server. It also allows integration with third-party modules and therefore enables flexibility in development [[44](" \l "Net15)].

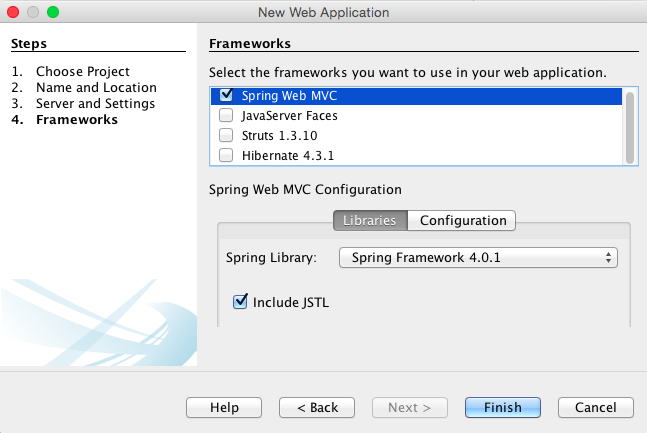


Figure 12: Creation of Spring Web MVC application in NetBeans

GitHub is a Git repository and it was used for review version control and management of the code. It was also used in order to share the code with the project supervisor and show progress thought-out the development phase of the application.

## Implementation plan

The main resource that the development plan was based on is the final year student - developer (CK) and the supervisor (SW) that supports and guides the project.

The web application was developed following the system architecture. The structure is separated in several layers to achieve efficiency and maintainability of the website. The 4 layers of system architecture are described as follows:

## Presentation layer

The Presentation layer is responsible for the (i) View and (ii) the presentation of data.

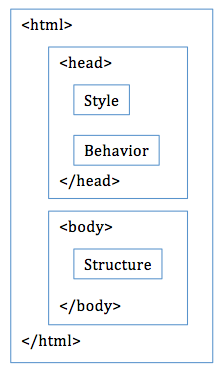
1. The View, which is also known as JSP pages in Web MVC terminology, generates responses and forwards them to the controller. The controller than sends the response to the client.
2. The presentation of data is archived by HTML5 and CSS. In order to ease maintenance and achieve maximum efficiency, the Style, Behaviour and Structure are clearly separated. The Style includes links of imported style sheets or external CSS files. The Behaviour includes links to local script elements or imported JavaScript files. Finally, the Structure includes the HTML structural elements.

Figure 10: Separation of style, behavior and structure

Bootstrap was also used for presentation purposes. Bootstrap is the most popular front-end framework for web development. The HTML and CSS templates that Bootstrap offers were customized and used to transform Pinboard into a responsive and dynamic web application. This allows the content of Pinboard web app to be rented on any device (desktop, mobile phone, tablet) regardless of the screen size or Operating System [[45](#aMa15)]. For example, when the size of the browser window is resized, the content scales proportionally to fit the window.

## Security Layer

The security layer deals with security concerns and it’s implemented in layers using a number of technological controls. First, the use of the University of Surrey email address restricts access only to students and staff of the University. Then, upon login, Spring MVC framework libraries are used for authentication of the user and authorization to access the website. Finally, Hyper Text Protocol Secure (HTTPS) is used as a communication protocol for secure communication over the network.

*Security defence – SQL injection:*

Login is required to enable users view the list of available items, create favourites and add new items for sale. During the login process the students and staff provide their university username (email) and password. The submission of their personal information generates an SQL query [[46](" \l "Bey14)]. The query is then executed on the database and the user’s account is authenticated and authorised to allow access to the website.

Text submitted by the user is always treaded as suspicious thus not only client side but also server side validation was applied. Lack of validation will allow cybercriminals to submit specifically crafted SQL commands (known as SQL injection) aiming to attack the application and reveal information about the database’s structure such as numbers and names of tables. The knowledge of such information will allow them to manipulate the tables, create admin accounts to manage the database and access users’ private information. In order to defend such operations from attackers, the SQL query is parameterised to prevent execution of malicious code.

*Form validation:*

In addition to parameterised SQL queries, a jQuery validator plugin is used to validate the input submitted by users through the login form. The customisation options offered by the plugin provide easy validation of input content. These include ready-made validation methods such as email validation and delivery of prepared error messages [[47](" \l "Jqu14)].

Validation is applied both on client and server side. The Client side validation offers a smooth user experience and it helps reduce the load on server by identifying user errors. The Server side validation checks the data submitted by the user to prevent SQL injection attacks of cybercriminals attempts to submit untrustworthy data.

## Business Logic

The Business logic consists of Spring MVC framework controllers and services. It receives the incoming information, performs specific operations and defines the content that needs to be returned to the user ensuring that the required functionalities are always available.

## Database Layer

*Database creation and integration:* The chosen relational database management system (RDBMS) is MySQL. The Structured Query Language (SQL) is used for the creation of tables and fields within the database and the establishment of primary and foreign keys.

MySQL Workbenchis also used as a visual database design and modeling tool. It was used for (i) design and generation of the database (ii) development of the SQL statements and (iii) configuration the server, user and admin permissions for Pinboard application [[48](" \l "Ora151)].

Apache Tomcat is an open source HTTP web server and Java based servlet container. It was used for the implementation of Java Servlets and Java Server Pages (JSP) specifications.

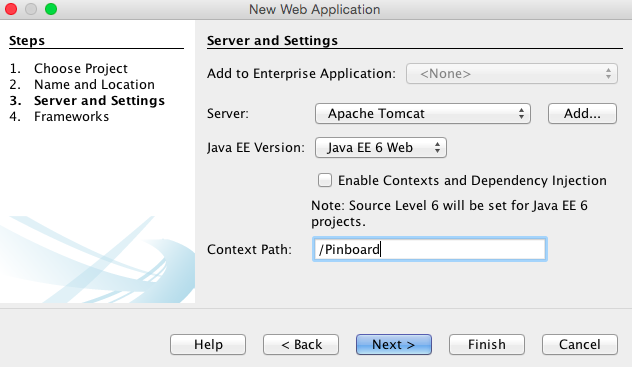


Figure 12: Apache Tomcat server set up in NetBeans

Connect to MySQL database using Java: <http://mrbool.com/how-to-connect-with-mysql-database-using-java/25440>

Details about remote connection to the database can be found in section 9.2 Appendix.

***Security Layer:***

* *Protection against CRSF attacks:*

(CK to be completed)

<!-- Protection against CRSF attack.

Hackers use Cross-Site Request Forgery attack in order to steal the cookies from the authenticated user.

Docs.spring.io, 'Spring Security Reference', 2015. [Online]. Available: http://docs.spring.io/spring-security/site/docs/3.2.x/reference/htmlsingle/#csrf-using. [Accessed: 09- Feb- 2015].-->

Software development:

* Spring MVC – Java
* JavaScript
* CSS3 is used to enable the website adapt to different screen sizes and resolutions whether rendered on mobile phone devices, tablets or desktops.

Libraries:

* Spring libraries for security
* Here add the rest of the sucuirty and spring libraries (jars):
  + Spring Security - jdbc authorisation: <http://www.mkyong.com/spring-security/spring-security-form-login-using-database/>
  + Spring Security - getting currently logged in user: <http://www.mkyong.com/spring-security/get-current-logged-in-username-in-spring-security/>
  + Spring Security - encoding password: <http://www.mkyong.com/spring-security/spring-security-password-hashing-example/>
  + Bootstrap CSS Library: <http://getbootstrap.com/>, <http://www.bootstrapcdn.com/>
  + Smooth back to top scrolling: <http://developerdrive.developerdrive.netdna-cdn.com/wp-content/uploads/2013/07/scroll-to-top.html>
* Passwords:
  + Hashing examples: <http://www.mkyong.com/java/java-sha-hashing-example/>
  + Best way to store passwords: <http://stackoverflow.com/questions/14798275/best-way-to-store-passwords-in-mysql-database>

## Success criteria

* The deliverables and software development requirements met.
* The objectives are met.
* The website is functional.
* Delivery of final project in agreed timescale.
* System testing (requirements and UAT) is documented and completed.

# Test and evaluation

## Requirements testing



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Req. no.** | **Requirements Description** | **Req. Type** | **Shall/Must** | **MoSCoW** | **Pass/Fail** | **Test notes** |
| **Database** | | | | | | |
| 1 | The centralized database that contains the products must update automatically and in real time. | Non-functional | Shall | M |  |  |
| 2 | The system must update the list of products so that a product becomes unavailable after a user purchases it. | Functional |  | M |  |  |
| **System permissions and roles** | | | | | | |
| 3 | The system must be available only to the University of Surrey students and staff. | Functional |  | M |  |  |
| 3.1 | The system must offer 2 different permission roles. One for admin-role and one for standard user-role. | Functional |  | M |  |  |
| 3.2 | The system must allow admins full control over the website, including deletion of user accounts and items. | Functional |  | M |  |  |
| 4 | The system must forbid standard-users from accessing admin pages. | Functional |  | M |  |  |
| Login - Logout | | | | | | |
| 5 | The system must forbid access to users with invalid username. | Functional |  | M |  |  |
| 6 | The system must provide a ‘Forgot my password’ functionality to allow users reset their password. | Functional |  | M |  |  |
| 7 | The system must provide logout functionality. | Functional |  | M |  |  |
| 8 | The system won’t require the users to register. The login process will be based on the existing UoS Active Directory (AD) groups. New students will be automatically added in the AD group and they will be able to access the website. | Functional |  | W |  |  |
| Validation | | | | | | |
| 9 | The system must validate the user information upon login. | Functional |  | M |  |  |
| 9.1 | A UoS Username must be provided in order to login. | Functional |  | M |  |  |
| 9.2 | Password must be provided in order to login. | Functional |  | M |  |  |
| 9.3 | Password must be at least 8 characters long. | Functional |  | M |  |  |
| 9.4 | Password must be hashed in the database. | Functional |  | M |  |  |
| 9.5 | The system must check if the username is valid by checking the list of usernames in the UoS AD group. | Functional |  | M |  |  |
| 9.6 | The system must decline usernames that are not valid. | Functional |  | M |  |  |
| 9.7 | The username must be 7 characters long. The first 2 characters must be letters.  Characters 3 to 7 must be numbers. | Functional |  | M |  |  |
| Users | | | | | | |
| 10 | The system should provide a ‘My profile’ area for each user. The personal information of the user will be shown on that page (Name, Contact details, email address, telephone number). | Functional |  | C |  |  |
| Items | | | | | | |
| 11 | The system must provide information about the product: Title, Description, Price, Category and Seller contact information. | Functional |  | M |  |  |
| 12 | The system must ask the seller to fill specific fields regarding the product: Title, Description, Price, Category and Seller contact information such as email address and/or mobile phone number. | Functional |  | M |  |  |
| Sort & Search items | | | | | | |
| 13 | The system must allow the users to search products based on categories e.g. books or room swaps. | Functional |  | M |  |  |
| 14 | The system must provide a search functionality (text-box search) to allow the users enter keywords. | Functional |  | M |  |  |
| 15 | The system could enable the users to sort the products based on price (Low to High, or High to Low). | Functional |  | C |  |  |
| Favourites | | | | | | |
| 12 | The system should allow users to bookmark items as ‘Favourites’. | Functional |  | M |  |  |
| 13 | Items marked as Favorites must be protected from external unauthorized access by other users. | Non-functional |  | M |  |  |
| **Terms & Conditions** | | | | | | |
| 3 | The system must ask the users (byers & sellers) to agree with ‘The Sales of Goods Act 1979’ and adhere to it: “*"traders must sell goods that are as described and of satisfactory quality”* [[28](" \l "Leg15)], [[33](" \l "Gla15)]. The Terms & conditions must be accessible, meaningful and fair. | Non-functional |  | M |  |  |

## Security testing

To be completed

## UAT testing

To be completed

# Conclusion

## Achievements, review of objectives

To be completed

## Issues and shortcomings

To be completed

A number of issues made the development of the website a challenging task.

* *Use of Spring Web MVC Framework:* The use of Spring MVC with no previous experience is a steep learning curve.
* *SHH Tunnel:* Connectivity issues due to SHH Tunnel problems from local workstation to remote server. Support was provided by the IT Services team and the project supervisor to overcome this issue.
* *User’s profile:* The user’s profile must be kept secure to ensure that no other users can gain access to it and modify sensitive information.

## Future work and suggestions for improvement

To be completed

* Additional features can be added to extend the list of available items offering a number of browsing categories like: unwanted tickets for events, electronics, sports equipment, office equipment like second hand bookshelves and drawers etc..
* A new page/section can be created to show the most recent items on the website and help the users stay up-to date with latest items available.
* Zoom feature can help the users get a closer look at the item and therefore enrich the user experience.
* PayPal services can be provided to ease the payment options. The initial system didn’t include PayPal services because students live on campus and the buyer and seller can meet for the delivery and payment of the item.

## Final evaluation and conclusions

To be completed

## Summary

To be completed

# Bibliography

(CK: delete the word hyperlink and add the date accessed too)

x

|  |  |
| --- | --- |
| [1] | Reuters. (2015) Most of world interconnected through email and social media'. [Online]. "http://www.reuters.com/article/2012/03/27/uk-socialmedia-online-poll- idUSLNE82Q02120120327 " http://www.reuters.com/article/2012/03/27/uk-socialmedia-online-poll- idUSLNE82Q02120120327 |
| [2] | The Radical Group. (2015) Most of world interconnected through email and social media. [Online]. "http://www.radicati.com/wp/wp-content/uploads/2013/04/Email-Statistics- Report-2013-2017-Executive-Summary.pdf" http://www.radicati.com/wp/wp-content/uploads/2013/04/Email-Statistics- Report-2013-2017-Executive-Summary.pdf |
| [3] | Esecurityplanet.com. (2015) Esecurityplanet.com. [Online]. "http://www.esecurityplanet.com/network-security/almost-100-billion- spam-e-mails-sent-daily-in-q1-2013.html" http://www.esecurityplanet.com/network-security/almost-100-billion- spam-e-mails-sent-daily-in-q1-2013.html |
| [4] | A. Osterwalder, Y Pigne, G Bernarda, and Osterwalder Smith. |
| [5] | SearchSoftwareQuality. (2015) Software development life cycle phases, iterations, explained step by step. [Online]. "http://searchsoftwarequality.techtarget.com/answer/Software- development-life-cycle-phases-iterations-explained-step-by-step " http://searchsoftwarequality.techtarget.com/answer/Software- development-life-cycle-phases-iterations-explained-step-by-step |
| [6] | Wikipedia. (2015) Comparison of free software e-commerce web application frameworks. [Online]. "http://en.wikipedia.org/wiki/Comparison\_of\_free\_software\_e- commerce\_web\_application\_frameworks " http://en.wikipedia.org/wiki/Comparison\_of\_free\_software\_e- commerce\_web\_application\_frameworks |
| [7] | Ofbiz.apache.org. (2015) Apache OFBiz, The Apache Open For Business Project - Open Source E- Business / E-Commerce, ERP, CRM, POS, SCM, MRP, CMMS/EAM. [Online]. "https://ofbiz.apache.org/ " https://ofbiz.apache.org/ |
| [8] | Oscommerce. (2015) Creating Online Stores Worldwide | osCommerce. [Online]. "http://www.oscommerce.com" http://www.oscommerce.com |
| [9] | Magendo. (2015) HP - Magento.com UK. [Online]. "http://magento.com/UK" http://magento.com/UK |
| [10] | Wikipedia. (2015) Gumtree. [Online]. "http://en.wikipedia.org/wiki/Gumtree" http://en.wikipedia.org/wiki/Gumtree |
| [11] | Secure.preloved.co.uk. (2015) About Us | Preloved. [Online]. "http://www.preloved.co.uk/about " http://www.preloved.co.uk/about |
| [12] | Wikipedia. (2015) [Online]. "http://en.wikipedia.org/wiki/Preloved" http://en.wikipedia.org/wiki/Preloved |
| [13] | Secure.preloved.co.uk. (2015) https://secure.preloved.co.uk/eu?qs=JfkBZrgTL0Us\_mTRoTIqPfCxFFVK68QIgWR8O2F9jgrDlqT\_ SK8RQQ. |
| [14] | Londonlovesbusiness.com. (2015) Londonlovesbusiness.com. [Online]. "http://www.londonlovesbusiness.com/entrepreneurs/famous-entrepreneurs/the-interview-the- mysterious-founders-of-gumtreecom/2210.article " http://www.londonlovesbusiness.com/entrepreneurs/famous-entrepreneurs/the-interview-the- mysterious-founders-of-gumtreecom/2210.article |
| [15] | Statista. (2015) Statista. [Online]. "http://www.statista.com/statistics/242235/number-of-ebays-total-active-users/ " http://www.statista.com/statistics/242235/number-of-ebays-total-active-users/ |
| [16] | Preloved.co.uk. (2015) Preloved.co.uk. [Online]. "http://www.preloved.co.uk/about " http://www.preloved.co.uk/about |
| [17] | Oracle. (2015) Oracle.com. [Online]. "http://www.oracle.com/technetwork/articles/java/mvc-2280472.html " http://www.oracle.com/technetwork/articles/java/mvc-2280472.html |
| [18] | Slideshare. (2015) Slideshare.net. [Online]. "http://www.slideshare.net/delabassee/mvc-10-jsr-371 " http://www.slideshare.net/delabassee/mvc-10-jsr-371 |
| [19] | D. MVC. (2015) Stackoverflow.com. [Online]. "http://i.stack.imgur.com/Discf.jpg" http://i.stack.imgur.com/Discf.jpg |
| [20] | Docs.spring.io. (2015) Spring. [Online]. "http://docs.spring.io/spring-framework/docs/3.0.x/reference/overview.html. " http://docs.spring.io/spring-framework/docs/3.0.x/reference/overview.html. |
| [21] | Spring.io. (2014) Spring. [Online]. "http://spring.io/ " http://spring.io/ |
| [22] | Spring.io. (2015) 9. Aspect Oriented Programming with Spring. [Online]. "http://docs.spring.io/spring/docs/current/spring-framework-reference/html/aop.html" \l "aop-introduction" http://docs.spring.io/spring/docs/current/spring-framework-reference/html/aop.html#aop-introduction |
| [23] | Spring.io. (2015) 2. Introduction to the Spring Framework'. [Online]. "http://docs.spring.io/spring/docs/current/spring-framework-reference/html/overview.html " http://docs.spring.io/spring/docs/current/spring-framework-reference/html/overview.html |
| [24] | Withoutbook.com. (2015) Withoutbook.com. [Online]. "http://www.withoutbook.com/DifferenceBetweenSubjects.php?subId1=27&subId2=28&d=Difference +between+Struts+and+Spring " http://www.withoutbook.com/DifferenceBetweenSubjects.php?subId1=27&subId2=28&d=Difference +between+Struts+and+Spring |
| [25] | Docs.Spring.io. (2015) 1. Introduction to Spring Framework. [Online]. "http://docs.spring.io/spring-framework/docs/3.0.x/reference/overview.html" http://docs.spring.io/spring-framework/docs/3.0.x/reference/overview.html |
| [26] | Coplien, T. (2015) Artima.com. [Online]. "http://www.artima.com/articles/dci\_vision.html " http://www.artima.com/articles/dci\_vision.html |
| [27] | D. Vrusias. (2015) University of Surrey, Surrey Learn. [Online]. HYPERLINK "https://sharepoint.surrey.ac.uk/exampapers/Papers/Advanced%20Challenges%20in%20Web%20Tech nologies.pdf " https://sharepoint.surrey.ac.uk/exampapers/Papers/Advanced%20Challenges%20in%20Web%20Tech nologies.pdf |
| [28] | Legislation.gov.uk. (2015) Legislation.gov.uk. [Online]. HYPERLINK "http://www.legislation.gov.uk/ukpga/1979/54 " http://www.legislation.gov.uk/ukpga/1979/54 |
| [29] | Sqa.org.uk. (2015) Functional and Non-Functional Requirements. [Online]. HYPERLINK "http://www.sqa.org.uk/e-learning/SDM03CD/page\_02.htm " http://www.sqa.org.uk/e-learning/SDM03CD/page\_02.htm |
| [30] | ReQtest. (2015) ReQtest. [Online]. HYPERLINK "http://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/ " http://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/ |
| [31] | Ietf.org. (2015) Ietf.org. [Online]. HYPERLINK "Available: http://www.ietf.org/rfc/rfc2119.txt. " Available: http://www.ietf.org/rfc/rfc2119.txt. |
| [32] | S Famuyide and S Famuyide. (2013) Business Analyst Learnings. [Online]. HYPERLINK "http://businessanalystlearnings.com/ba- techniques/2013/3/5/moscow-technique-requirements-prioritization " http://businessanalystlearnings.com/ba- techniques/2013/3/5/moscow-technique-requirements-prioritization |
| [33] | Glassraven.com. (2015) Glassraven.com. [Online]. HYPERLINK "http://www.glassraven.com/articles/online-legal-requirements.php" http://www.glassraven.com/articles/online-legal-requirements.php |
| [34] | Extension.iastate.edu. (2015) Extension.iastate.edu. [Online]. HYPERLINK "http://www.extension.iastate.edu/agdm/wholefarm/html/c5-65.html " http://www.extension.iastate.edu/agdm/wholefarm/html/c5-65.html |
| [35] | Dictionary.com. (2015) Dictionary.com. [Online]. HYPERLINK "http://dictionary.reference.com/browse/stakeholder " http://dictionary.reference.com/browse/stakeholder |
| [36] | stqbexamcertification.com. (2015) stqbexamcertification.com. [Online]. HYPERLINK "http://istqbexamcertification.com/what-is-waterfall-model-advantages- disadvantages-and-when-to-use-it/" http://istqbexamcertification.com/what-is-waterfall-model-advantages- disadvantages-and-when-to-use-it/ |
| [37] | Cs.odu.edu. (2015) Cs.odu.edu. [Online]. HYPERLINK "http://www.cs.odu.edu/~cs333/website-s11/Lectures/waterfall/page/waterfall.html" \l "subsubsect-design" http://www.cs.odu.edu/~cs333/website-s11/Lectures/waterfall/page/waterfall.html#subsubsect-design |
| [38] | Docs.spring.io. (2015) Spring.io. [Online]. HYPERLINK "http://docs.spring.io/spring/docs/current/spring-framework-reference/html/new-in- 4.0.html " http://docs.spring.io/spring/docs/current/spring-framework-reference/html/new-in- 4.0.html |
| [39] | Pages.zeroturnaround.com. (2014) Pages.zeroturnaround.com. [Online]. HYPERLINK "http://pages.zeroturnaround.com/Java-Tools- Technologies.html?utm\_source=Java%20Tools%20&%20Technologies%202014&utm\_medium=allr eports&utm\_campaign=rebellabs&utm\_rebellabsid=88 " http://pages.zeroturnaround.com/Java-Tools- Technologies.html?utm\_source=Java%20Tools%20&%20Technologies%202014&utm\_medium=allr eports&utm\_campaign=rebellabs&utm\_rebellabsid=88 |
| [40] | Docs.spring.io. (2015) Docs.spring.io. [Online]. HYPERLINK "http://docs.spring.io/autorepo/docs/spring-security/4.0.0.CI- SNAPSHOT/reference/htmlsingle/" \l "appendix-dependencies " http://docs.spring.io/autorepo/docs/spring-security/4.0.0.CI- SNAPSHOT/reference/htmlsingle/#appendix-dependencies |
| [41] | H Maylor, *Project management* ,. ed.,., Ed. England, Harlow : Financial Times Prentice Hall, 2010. |
| [42] | Team and A. Profile. (2015) Twago.com. [Online]. HYPERLINK "http://www.twago.com/blog/project-plan-makes-life-easier/ " http://www.twago.com/blog/project-plan-makes-life-easier/ |
| [43] | Barnes. (2009) Smashing Magazine. [Online]. HYPERLINK "http://www.smashingmagazine.com/2009/06/11/effective-strategy-to-estimate-time-for-your-design- projects/ " http://www.smashingmagazine.com/2009/06/11/effective-strategy-to-estimate-time-for-your-design- projects/ |
| [44] | Netbeans.org. (2015) NetBeans IDE - Overview. [Online]. HYPERLINK "https://netbeans.org/features/index.html" https://netbeans.org/features/index.html |
| [45] | Oracle.com. (2015) Oracle.com. [Online]. HYPERLINK "http://www.oracle.com/us/products/mysql/mysql-workbench-066221.html" http://www.oracle.com/us/products/mysql/mysql-workbench-066221.html |
| [46] | Beyondsecurity.com. (2014) Beyondsecurity.com. [Online]. HYPERLINK "http://www.beyondsecurity.com/about-sql-injection.html " http://www.beyondsecurity.com/about-sql-injection.html |
| [47] | Jqueryvalidation.org. (2014 ) Jqueryvalidation.org. [Online]. HYPERLINK "http://jqueryvalidation.org/ " http://jqueryvalidation.org/ |
| [48] | Info.eps.surrey.ac.uk. (2015) Info.eps.surrey.ac.uk. [Online]. HYPERLINK "http://info.eps.surrey.ac.uk/IT/guides/remoteaccess/linuxSshTunnel.php " http://info.eps.surrey.ac.uk/IT/guides/remoteaccess/linuxSshTunnel.php |
| [49] | Blog.trackets.com. (2015) Trackets Blog. [Online]. HYPERLINK "http://blog.trackets.com/2014/05/17/ssh-tunnel-local- and-remote-port-forwarding-explained-with-examples.html. " http://blog.trackets.com/2014/05/17/ssh-tunnel-local- and-remote-port-forwarding-explained-with-examples.html. |

x

# Appendices

## Response from the IT Services.

Macintosh HD:Users:CharaKatiri:Documents:Pinboard:COM3001_Documentation:IT emails:ResponseFromITSupport.pdf

## Creation and remote connection to the database behind the firewall

Due to security reasons, direct connection to a service (database console), hosted on a protected network (University of Surrey network), is not possible [[49](" \l "Inf15)]. The steps followed to enable the connection are described below:

1. Creation of a remote server on the protected network as well as a local workstation (a local MySQL server, database). Following the completion of the development phase on the (i) local workstation, the Pinboard website is deployed on the (ii) remote server.
2. To establish the connection between the (i) local workstation and the (ii) remote server a tunnel that uses a secure protocol is created.
3. Access to a free port can only be achieved from the localhost and not remotely [[50](" \l "Blo15)]. For security reasons a high, local, port such as 10,000 is used and PuTTY listens to it for incoming connections.
4. The MySQL database that runs on the server listens to port 3306.

$ ssh ck00113@student01.eps.surrey.ac.uk -L 10000:mysql0.ee.surrey.ac.uk:3306

The use of ‘-L’ states that the port is being forwarded. The use of ‘10000:mysql0.ee.surrey.ac.uk:3306’ forwards the connections from the local port (10000) to the server port (mysql0.ee.surrey.ac.uk:3306). During this step a connection (Tunnel) is made between the local port and the server port.

1. Following the port set-up, the details for connection are pulled from the config file and the connection to the remote server is established.
2. The use of ‘mysql -h mysql0.ee.surrey.ac.uk -u ck00113 -D ck00113 -p ck00113’ enables the connection to the database. This process needs to be followed only when the developer accesses the server off-campus.

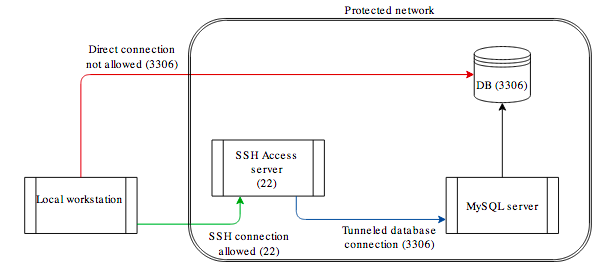


Figure 5: SSH Tunnel

## Project Plan

