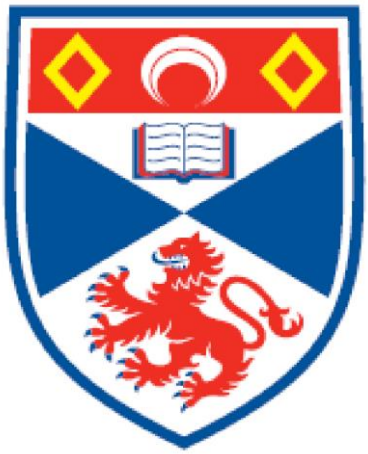
University of St Andrews

School of Computer Science



Scientific Data Management System for

Powder X-Ray Diffraction

Anke Shi
Matriculation Number: 160025235

MSc. Software Engineering
14 August 2017

Abstract

In this time of Information Technology, almost every detail of our lives has been taken over by web applications, still, some traditional manual booking systems are still in use. As demand increases, it is a historical necessity that intelligent and efficient web-based management tools replace these old-fashioned system. The project undertaken takes Powder X-Ray Diffraction (PXRD) as a real use-case to develop an online booking application. This dissertation describes the methodology, requirement, design, implementation and evaluation of this application. Based on the needs of PXRD, this application provides some basic but vital functionalities to transform the manual process into an electronic one.

Declaration

I hereby certify that this dissertation, which is approximately <N> words in length, has been composed by me, that it is the record of work carried out by me and that it has not been submitted in any previous application for a higher degree. This project was conducted by me at The University of St Andrews from 05/2017 to 08/2017 towards fulfilment of the requirements of the University of St Andrews for the degree of MSc under the supervision of Tom Kelsey and Shyam Reyal.

In submitting this project report to the University of St Andrews, I give permission for it to be made available for use in accordance with the regulations of the University Library. I also give permission for the title and abstract to be published and for copies of the report to be made and supplied at cost to any bona fide library or research worker, and to be made available on the World Wide Web. I retain the copyright in this work.

Date: 14/08/2017 Signature:

Acknowledgements

I would like to express my sincere gratitude to my supervisors Tom Kelsey and Shyam Reyal to offer this project, the kind support and guidance. I would also like to thank Simone Conte for his generous help.

Special thanks to my parents and friends for their love and support. Thanks to all the people I met this year. It is my pleasure to meet you!

Contents

List of figures:

List of Acronyms

One-button booking: One-button booking mode add booking by one button
Calendar booking: Calendar booking mode make booking by selecting form calendar
Asset-booking: Asset-booking mode needs the disc name, group and user name, range, and duration information
PXRD: Powder X-Ray Diffraction
MS: Mass Spectrometry
SS: Solid State NMR

Chapter 1 Introduction

1.1 Background

1.1.1 A brief overview of current systems

There are more than 20 different techniques supported by the School of Chemistry at the University of St Andrews. Among these, Powder X-Ray Diffraction (PXRD), Mass Spectrometry (MS), Solid State NMR (SS) and X-Ray Crystallography use a paper-based manual booking system as their reservation mechanism, while NOMAD is used by solution-state NMR spectrometry. The aim of this project is to provide the services of the NOMAD system, currently limited to NMR, to other techniques across the School of Chemistry.

There are 6 different working machines for Powder X-Ray diffraction to assist a number of research groups with material identification and other analysis. PXRD takes three different booking methods to arrange the experiment – Asset, Calendar and One-button, as named by the developers. They are varied in terms of operation modes and the booking duration that users require. Asset-booking takes the disc name, group and user name, range, and duration on the booking sheets (Appendix B). By the machine limitation, the administrator examines the viability of booking manually. The calendar-booking needs the user name and group name details for the required time periods. The one-button booking mode is activated with user details just before using the machine.

1.1.2 The NOMAD system

NMR Online Management And Datastore (NOMAD) system was created in 2012 to manage Nuclear Magnetic Resonance (NMR) data in collaboration between the Schools of Chemistry and Computer Science. It provides the experiment booking, tracking machine and experiment status, the archival, storage, retrieval and manipulation of NMR spectrum data and the generation of financial reports for over 600 users. By the introduction of NOMAD, the old paper based booking system was replaced with an online service.

1.2 Objectives

1.2.1 Primary objectives

- General requirement gathering and elicitation from multiple departments in chemistry, including Powder X-Ray diffraction
- The generic model design and identification of differences between different machineries
- Software implementation for calendar and one-button booking modes in PXRD
- Client's and users' evaluation of PXRD booking system

1.2.2. Secondary Objectives

- Software implementation for assets-booking mode
- Adapting calendar and one-button booking modes to other machineries
- Evaluation of booking system for other machineries

1.2.3 Tertiary Objectives

- UX/UI analysis and implementation
- Investigating booking machinery using mobile message bots

1.3 Proposed solution and contributions

This dissertation aims to solve the practical need of actual clients by using standard software engineering methodologies as practiced in the industry. Considering the needs of services and software quality, a Web application based on NOMAD was proposed to manage reservation and data for Powder X-Ray Diffraction with the following functions:

- Machine booking for standard users and admin users considering usage limitation
- Booking and resource management
- Data collection, i.e. getting group usage reports

The project makes the following contributions:

- System provides services with high accessibility and meets the need for concurrency. Users could break the constraints of time and space.
- Administrator adds new machines and updates machine settings easily with visualized operations.

- The scientific counting method eliminates the manual calculation.
- NOMAD team could take over the maintenance of software and database.

1.4 Ethical considerations

In this project, there is no sensitive data collected from human subjects during the project requirements phase, nor when obtaining feedback about the artifact. All data collection is performed under the approval code CS12476. The preliminary ethics self-assessment form and the artifact evaluation form can be found at Appendix A.

1.5 Report outline

This report structure follows the entire software development life cycle. Firstly, it introduces the current booking schema and the objectives of this project (Chapter 1). Then the context survey (Chapter 2) and methodology (Chapter 3) introduces the background from business and technical viewpoints. Chapter 4 shows the result generated from the requirement engineering process. Chapters 5 and 6 use examples to explain the design and implementation of the final system. Chapter 7 gives the evaluations from users. Finally, Chapter 8 sums up the work done as a conclusion, and states possible future work.

Chapter 2 Context survey

2.1 Booking systems

2.1.1 Introduction

If we want to travel for a while, a room needs to be booked to sleep; if we want to go to a popular restaurant, a table needs to be booked; if we want to use a public computer or a machine, a time slot needs to be booked. When the reservation concept is accepted by the public as a common sense, different kinds of effective booking systems are brought to give an open and fair reservation schema.

In general, the administrator sets up the available tickets while users can select by needs. The given input data are then delivered as requests. Then the arrangement could be made manually and automatically. The users are notified with the result. Due to the variety of user needs, booking systems have different kinds of workflow and techniques. Because of the complexity of booking rules, the paper-based booking system is still widely used to make reservations. As the name suggests, the paper-based system takes the booking records on the paper, while online booking uses digital management and storage. A shared document booking mode is the simplest way to replace the paper usage. A central reservation system is more intelligent as it provides various functionalities. The design concept of calendar booking has gained its popularity as many booking systems have a dynamic calendar reservation component.

2.1.2 Existing booking system analysis

2.1.2.1 Scientific systems

Research and scientific facilities, both in academia and industry, provide researchers with available resources to conduct scientific research work. Due to the diversity of experimental equipment, the usage of instruments is limited by complexity and lack of communication between different laboratories, as well as the strict rules of experiments and opening hours. Therefore, booking systems are designed accordingly to help maximize the utilization of instruments. With the limitation of resources and the growing reliance on the Internet, opening an online management tool would maximize

the utilization of laboratories, varying from online website to a mobile application (W. et al, 2012).

Shared remote online laboratory resources are a new hot issue raised by the development of distance education. The virtual experiments bring traditional on-campus experiments with challenges. For the input data, the estimation of time is still requested. However, the unique scheduling system and user performance should be taken into consideration while the institution boundaries are broken (Li, Esche & Chassapis, 2008). The synchronization of request and priority schema shall customize into online reservation protocol, for example, the live-class experiments have higher priority than undergraduate student tutorial test (Li, Esche & Chassapis, 2008). Some applications have already been used and reported to the public. For examples, one online laboratory uses one central system to support all online requests (De Vora, Auer & Grout, 2007). A Moodle extension management tool with immediate application takes user booking to access the MARVEL labs (Ferreira & Cardoso, 2005)

The public service reservation is different from others because of the greater number of users. Due to the diversity of user backgrounds and urgent needs to access the same resource, a booking system provides an efficient and highly simplified reservation process. The designs of user interfaces, input data and charges criteria need to cater to customer psychology and the user behaviour patterns (Teuber,C. & Forbrig P., 2004). The synchronization requests problem should be properly handled. For example, iTelescope.net offers over 10,000 users with an Internet-connected telescopes network to conduct observation[1]. A dynamic calendar reservation is used to show available time slots, discounts and other relevant information and manage reservation through dragging and click. The limited full-dark period, illumination and experiment plan give this booking system special characteristics.

2.1.2.2 Non-scientific booking systems

The booking systems play an essential role in our daily life, as we use them everywhere and all the time, such as the hotel booking, airline booking, restaurant booking,etc. The hotel industry has gone through a dramatic change over the last decades as the internet takes over the traditional communication (McTavish & Sankaranarayanan, 2010). Instead of calling and investigating yellow pages in advance, a comprehensive agent-based booking system has gained in popularity in recent years, such as booking.com, Airbnb.com, Hotels.com. An agent is an internal system, gathering information from its environment to chase its goal (McTavish & Sankaranarayanan, 2010). The hotel booking agents gather the hotel information as much as possible - location, prices, type of facilities, host information and rates, synchronize all the booking data and return the outcome through interacting with different service agents and resources. Unlike the

[1] http://www.itelescope.net/reservatio

scientific systems, it allows the user to book with just basic details, which is also included by other universal booking systems. Hotel booking systems also pay more attention to the accuracy of search results, as the search algorithms are more intelligent and goal-oriented than simple search (McTavish & Sankaranarayanan, 2010).

Nowadays, the airline booking has similar circumstance as hotel booking with an agent-based booking system. While the airline has its different attentions, it concerns mostly the seat occupancy (Knight, 1972). To make full use of available space, the booking requests the details from users to prepare the flights because of load factors and to arrange seats exhaustively for the disabled people and emergency safety issues (Knight, 1972). Airline booking systems also need an adequate searching algorithm and essential internal database manipulation to give a reasonable solution with minimum input data. Meanwhile, the involved authentication logic, accuracy of results, stability and reliability have higher requirements than other booking systems.

2.2 The school of Chemistry at University of St Andrews

The school of Chemistry at University of St Andrews offers more than 20 different kinds of facilities[2] to the researchers, including the Powder X-Ray Diffraction, Single Crystal X-ray diffraction, Solid-state NMR spectroscopy, Mass Spectrometry, etc. As part of EaStCHEM and ScotCHEM, it also allows academic and industrial users to access facilities. There are 6 different diffractometers serving around 100 users for Powder X-Ray Diffraction[3] – two PANalytical Empyrean with Cu X-ray tube (JOHN & PAUL), two Stoe Stadi p with Cu X-ray tube (GEORGE & RINGO), one PANalytical Empyrean with Mo X-ray tube (PETE) and one MINIFLEX 600. There are four different systems[4] to carry out the chemical crystallography search - STANDARD system, Rigaku Cu MM007 HF (dual port) high brilliance generator, Rigaku FRX (dual port) high brilliance generator and Rigaku SCX Mini. The 400 MHz HFXY and 600 MHz Bruker Avance III spectrometers explore the research on the NMR spectroscopy in solid state[5]. The school also runs two mass spectrometers[6] to conduct chemical and biomedical sciences research.

---

[2] https://www.st-andrews.ac.uk/chemistry/research/services/

[3] http://chemistry.st-andrews.ac.uk/pxrd/index.html

[4] https://www.st-andrews.ac.uk/chemistry/research/services/

[5] https://ssnmr.wp.st-andrews.ac.uk/available-equipment/

[6] http://mass-spec.wp.st-andrews.ac.uk/services/

Chapter 3 Methodology and Infrastructure

3.1 Requirements engineering

'Requirements' are the conditions or capabilities which a system or system component needs to satisfy (Jalote, 2005). In other words, it explains what the system should do and how the system should be implemented. The specifications of actions are classified as the functional requirements, while the system properties are treated as the non-functional requirements. The requirement engineering process refers to the elicitation, analysis, specification, validation and management of gathered requirements (Sommerville, 2016). The quality of software products mainly depends on the quality of the requirement engineering process (Pandey, Suman and Ramani, 2010). Therefore, it is vital to apply requirement engineering in every phase of development from different viewpoints and objectives (Pandey, Suman and Ramani, 2010).

In this project, the requirement engineering was carried out iteratively with consideration of technical, operational, legal and schedule feasibility. At the first meeting, the client (Dr Yuri Andreev) proposed the basic functionalities (see Appendix B). To achieve the reusability of software, the requirements from three other departments (Mass Spectrometry, Solid-State NMR, X-Ray Crystallography) were also gathered through meeting. Then the requirements were identified and documented by the researcher. The requirement verification and validation activities were carried under the supervision of the client and supervisors. At different stages of the software development life cycle, requirements were adjusted accordingly.

3.2 Iterative development

From Chapter 1 and 3.2, the project background and time limitation determines a rapid system development and process. Agile software development is an incremental software development method, which could produce software and satisfy stakeholders faster than the conventional waterfall process for business systems (Sommerville, 2016). The Agile method divides the software development process into several short periods named sprints (see Figure 1). There were four sprints (every 2 weeks) in this project. Each sprint started with the product backlog, given by the stakeholders and identified as requirements by the developer. Then, the developer chose missions to complete. A weekly meeting was held to discuss the progress and problems between advisors and the developer. After the sprint review, the potentially releasable product was presented

to the stakeholders. Alpha testing took place on the development platform to test its functionality by developer and client. Stakeholders gave suggestions and feedback from different aspects of the latest product. At the last sprint, Beta testing gave the common users an opportunity to voice their suggestions. A new sprint backlog was established with unfinished mission and feedback. The new sprints start until completion of the product backlog.

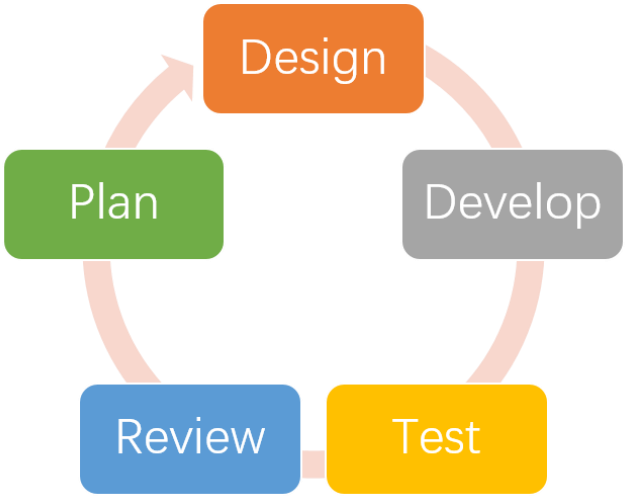


Figure 1 Agile methodology

3.3 Version Control

Rome wasn't built in a day, neither is this project. To back up files and keep track of different versions of programs, version control is a good solution to reduce unnecessary losses. As an extension of NOMAD, version control also prevents clashes with other components. To keep the consistency of sharing and synchronization strategy, Mercurial[7] was used as the distributed version control system, which is supported by the School of Computer Science in St Andrews[8]. Based on NOMAD repository, Mercurial provides the researcher a method to learn from its history of development by tracking the changes of files. Mercurial supports recording date, difference and author, reconstructing the previous state and branching to maintain different versions (Hinsen, Läufer and Thiruvathukal, 2009).

During the program development stage, feature branches were generated to record different numbered states and merged with success functions. One branch was maintained to record the well-functioned version. One branch was used to store the code even though there were a number of bugs and problems to be solved. One branch

[7] http://mercurial.selenic.com
[8] https://systems.wiki.cs.st-andrews.ac.uk/index.php/Mercurial_service

was created to place the trial with fullcalendar API, which were closed after this approach was abandoned (see Figure 2). Regular commits and pushes recorded the changesets, allowing the developer to update the local copy and allowing code governance and peer review as well.

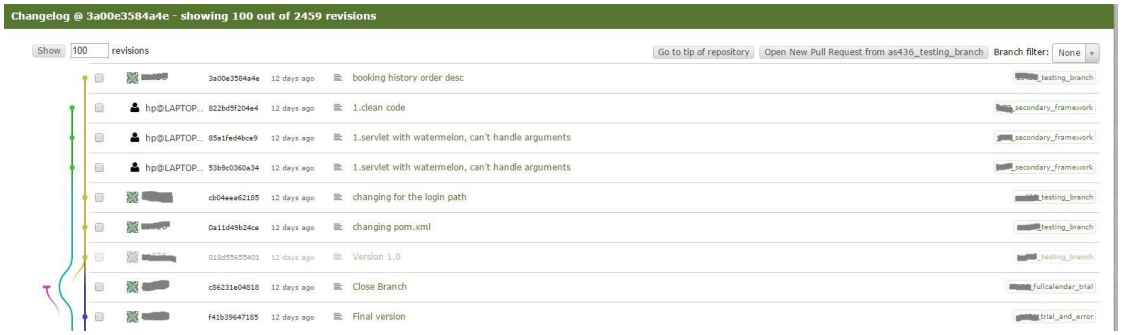


Figure 2 Changelog

3.4 JavaEE Architecture

JavaEE[9] is abbreviation for Java Platform Enterprise Edition, also called J2EE. It provides platform to develop enterprise software with high scalability, flexibility and accessibility. To avoid the shortcomings of two-tier (client/server) architecture, JavaEE adopts multitier strategy: Client tier, Web tier, Business logic tier and Enterprise information system tier. Java Servlet and Java Server Pages (JSP) are components from Web tier to support interactive browse, data modification and dynamic web content generation.

Observer (Listeners): When loading a web application, the web.xml (configuration file) loads ServletContext, context-param, listener, filter and servlet in sequence.The listener acts like an observer to listen to the application and trigger a response action. There are 8 different listeners which could be categorized into 3 different types by its object. For example, HttpSessionListener stores and destroys session events while HttpSessionAttributListener manages (includes adding, replacing and removing) the session attributes. HttpSessionActivationListener helps recover session information when the server restarts.

Front Controller (Servlet Filters): A servlet filter plays the role of intercepting the HttpServletRequest and HttpServletResponse objects and even modifying their heading and data. When the user sends requests or relevant URL, the servlet filter calls the doFilter () function first, in which the filter can execute specific codes or give

[9] http://www.oracle.com/technetwork/java/javaee/overview/index.html

permission to access requested resources. The filter-mapping maps with one or multiple Servlet or JSP to compress response time and filter sensitive vocabulary.

Façade Pattern (Meta Management and Configuration): In object-oriented architecture, façade is one of the most popular design patterns to drop coupling of code. The essence of the façade pattern is to isolate the subsystems from the client with a middle class. Façade knows functionalities of one or more subsystems, which are a set of classes. It takes requests from the client and appoints them into different subsystems. It provides a simplified Meta management method instead of creating new functions. Façade pattern slashes the complexity of subsystems and improves their independence.

Application Server (Apache Tomcat): The web server handles all HTTP (hypertext transfer protocol) requests, while the application server holds business logic. Apache is a web server and Tomcat [10] is a light application server to support Servlet/JSP. Developers used to combine Apache and Tomcat together to support their website. Apache takes HTTP requests from the client and then transfers requests from Servlet and JSP to Tomcat. After Tomcat finishes processing, it sends responses back to Apache, which finally send them to the client.

Apache Maven: Apache Maven[11] is a tool to easily manage and build projects. The pom.xml file is the kernel, which includes the dependencies, plugins, project version and other configuration information. During the build lifecycle, the validation, compilation, testing, packaging, integration-testing, verification, installation and deployment are conducted automatically phase by phase.

3.5 Project Management

3.5.1 Requirements Management

After clarifying the objectives, the development process was divided into 4 sprints, with each assigned a number of requirements. Before each sprint began, the advisors and developer outlined plans together. The sprint backlogs were updated on Trello (a project management tool) during the sprint. It visualized the information with cards labelled ‘to-do’, ‘doing’, ‘under-view’ and ‘done’. Through the whole sprint, cards were added or edited by the developer or advisors. The specific details were determined by the estimated time and size of the task.

[10] http://tomcat.apache.org/
[11] http://maven.apache.org/

3.5.2 Collaboration

To cooperate with the NOMAD team, Kallithea was used to do the code governance and peer review. It provides the files, branches and other version control histories. It allows the other team member to review and leave comments. Meanwhile, team communications are maintained through Slack, where NOMAD team members and clients could receive instant messages.

There are substantial collaborations throughout the project:

- Tom Kelsey gave suggestion on project management and supervised the project progress.
- Shyam Reyal helped organize the meeting with clients and gave advices on any problems during design, implement and report writing phases.
- Simone Conte also gave technical governance and participated in the system design, implement and report.
- NOMAD team acts as technical consultant.
- Dr Yuri Andreev and PXRD users tested system and shared their feedback.

Chapter 4 Requirement

The requirements in this chapter are mainly taken from the meeting with clients, including Dr Yuri Andreev (PXRD), Prof Alexandra Slawin (X-Ray diffraction), Dr Daniel McLean Dawson (Solid State NMR), Mrs. Caroline Horsburgh and Mrs. Sally Lorna Shirran (Mass Spectrometry). Most of the requirements were enumerated after the meeting through the discussion with the project supervisor Tom Kelsey and the PhD student supervisors Shyam Reyal and Simone Ivan Conte, who helped with the understanding of the system.

An initial list of potential clients was generated from the Chemistry school website and Prof David O'Hagan (Head of the School of Chemistry) gave the contact details of selected members from the list. The eventual four groups were chosen as the focus group because of their written reservation systems and needs. In order to apply the e-working booking system and NOMAD, the booking process and data retrieval process were questioned. Before each meeting, the interviewees were informed of the aim of interview. Through the meeting, the stakeholders declared their requirements and elaborated their current booking procedure with real cases. Then they were asked about the further details, urgency of a full-functional system and economic benefit. As the PXRD is the initial client group, the main requirements are generated for it. Moreover, the requirements for other three technique departments are listed by the order of workflow. At the end, the adaptability and changes of design are summarized in the comparison of requirements.

4.1 Universal requirements

The main purpose of having a scientific data management system is to organize the scientific life efficiently and to make data trackable and searchable. In general, for a functional booking system, it must meet the following requirements:

- The user shall be able to submit and upload relevant booking details.
- The user shall be able to make an appointment.
- The user shall be able to retrieve their booking history.
- The user shall be able to cancel these bookings following the regulation.
- The user (administrator) shall be able to manage data.
- System shall be available and accessible.

4.2 Powder X-Ray Diffraction (PXRD)

The first meeting was on the 15th of May, 2017 at the chemistry Purdie building, St Andrews. At the meeting, the client (Dr Yuri Andreev) demonstrated the instruments and current workflow at the PXRD lab. Then he explained stipulations and management demands for each type of booking exhaustively. The initial requirement paper and reservation form can be found in Appendix B. At the second meeting he gave feedback and suggestions about the user interface design and the requirements were changed accordingly. From the following demo and initial presentation meeting, the stakeholder considered the practicality of the system and the functional and non-functional requirements were adjusted. The stakeholders and developer contact continuously and promptly during the whole process. The precise functional and non-functional requirements are addressed at 5.2.1 and 5.2.2.

4.2.1 Functional requirements

The finalized functional requirements are divided by different booking mode aspects. The general requirements demonstrate the basic usage of the system. The Asset-booking, Calendar and One-button functional requirements specify the corresponding requests made by user and system. As the administrators are expected to understand the booking rule and manage system, the common functionality were reorganized with other management requirements deliberately.

4.2.1.1 General requirements

ID	Requirement	Importance
1	The user shall be able to log in with university credentials.	High
2	The user shall be able to book three different types of diffractometers: One-button booking (MINIFLEX), Calendar booking (George & Ringo) and Asset-booking (John & Paul).	High
3	The user shall be able to find their booking history for three different types of diffractometers.	High
4	The user shall be able to cancel his/her own booking within valid time.	High
5	The user shall be able to find his/her own current day booking agenda.	Medium
6	The user shall be able to log off the system.	High

The table lists the basic functionalities from daily usage with which online system could replace all operation from written reservation form.

4.2.1.2 Asset-booking functional requirements

ID	Requirement	Importance
7	The user shall be able to make a reservation with duration, range, holder and film details for next 2 working days.	High
8	The user shall not be able to book the experiment on the current day after a fixed time.	High
9	The user shall be able to choose to add protective film.	High
10	The user shall be able to see the booking history from last 7 days.	Medium
11	The user shall be able to cancel booking before the fixed time.	High
12	The user shall not be able to book when daily booking times have reached a maximum number of daily personal booking times.	High
13	The user shall not be able to book when selected diffractometer has reached maximum running hours (Monday to Thursday).	High
14	The user shall not be able to book when selected diffractometer has reached its Friday maximum running hour.	High
15	The user shall not be able to book when the total number of booked discs has reached its maximum on the selected date.	High
16	The system shall be able to record booking with user identity and selected details.	High
17	The system shall be able to provide the range choices by machine from archived program.	High
18	When the range is selected, the system shall be able to provide available duration from the existing combination	Medium
19	The system shall not be able to provide user with booked disc holder in the same day.	High
20	The system shall be able to show the success prompt window.	Medium
21	The system shall be able to show the failure prompt window when the operation violates the booking rule.	Medium

The table shows all requested functionalities in detail, which are expected to operate on the Asset-booking machine (John & Paul).

4.2.1.3 Calendar functional requirements

ID	Requirement	Importance
22	The user shall be able to book diffractometer by selecting from available time slots for next 7 days.	High
23	The user shall not be able to select from past time.	Medium

24	The user shall not be able to book when the number of personal daytime bookings has reached its maximum.	High
25	The user shall not be able to book when the number of personal weekly overnight bookings has reached its maximum.	High
26	The user shall be able to cancel their booking before the booked start time.	High
27	The user shall be able to see all booking records with user name and group name for next 7 days.	High
28	The system shall be able to record booking with user identity and selected time slot.	High
29	The system shall be able to show the success prompt window.	Medium
30	The system shall be able to show the failure prompt window when the operation violates the booking rule.	Medium

The table declares all requested functionalities for Calendar-booking machine (George & Ringo).

4.2.1.4 One-button functional requirements

ID	Requirement	Importance
31	The user shall be able to record experiment before using the machine with the disc number and comments (optional).	High
32	The user shall not be able to book while the time since the last booking is less than the specified machine running time.	High
33	The user shall be able to see his/her booking history.	High
34	The system shall be able to record booking with user identity, time and provided information.	High
35	The system shall be able to show the success prompt window.	Medium
36	The system shall be able to show the failure prompt window when the operation violates the booking rule.	Medium

The table concludes all necessary functionalities for MINIFLEX, One-button machine.

4.2.1.5 Admin functional requirements

ID	Requirement	Importance
37	The user shall be able to add new user.	High
38	The user shall be able to edit the existing user.	High
39	The user shall be able to inactivate/activate user.	Medium

40	The user shall be able to see all the user information.	High
41	The user shall be able to add new group.	High
42	The user shall be able to edit the existing group.	High
43	The user shall be able to inactivate/activate group.	Medium
44	The user shall be able to delete group.	High
45	The user shall be able to see all the group information.	High
46	The user shall be able to see all the current day booking ordered by the machine name and booking orders for asset-booking.	High
47	The user shall be able to see all the current day booking ordered by the machine name and timeslot for calendar-booking.	High
48	The user shall be able to add all three types of booking with username details under no restriction.	High
49	The user shall be able to cancel all three types of booking.	High
50	The user shall be able to choose all the holders.	High
51	The user shall be able to add new machines for all three types of booking.	High
52	The user shall be able to change the existing machine details and settings for all three types of booking.	High
53	The user shall be able to inactivate/active machine for all three type of booking.	High
54	The user shall be able to add new program for asset-booking diffractometer.	High
55	The user shall be able to change the existing program.	High
56	The user shall be able to add new holder for asset-booking diffractometer	High
57	The user shall be able to edit the existing holder.	High
58	The user shall be able to get the report of number of runs and hours for each group by machines during selected time period.	High
59	The user shall be able to edit the timeslot.	Medium
60	The user shall be able to get the report of number of bookings and overnight bookings for each group by machine during selected time period.	High
61	The user shall be able to add the disc, used by one-button machine.	High
62	The user shall be able to edit the disc details.	High
63	The user shall be able to search the booking record by time stamp.	Low
64	The user shall be able to get report of usage for each group during the selected time period.	High
65	The system shall be able to provide existing groups to select when editing user information.	High
66	The system shall be able to show the success prompt window.	Medium
67	The system shall be able to show the failure prompt window when the operation violates the booking rule.	Medium

The table summarizes the daily operation and maintenance needs for the admin.

4.2.2 Non-functional requirements

ID	Requirement	Importance
68	The system shall be robust to work.	High
69	The system shall be accessible by desired users.	High
70	The system shall protect the privacy of users.	High
71	The system shall be user-friendly and easily understood by non-computer background users.	High
72	The system shall be able to deliver the success/failure message.	Medium
73	The system shall be available 99.999% of time.	Medium
74	The system shall be maintainable and modifiable.	High
75	The system shall be reusable to others	Low
76	The system shall be developed to cohere with Hyper Text Markup Language (HTML) guidelines and standards.	High

4.3 Mass Spectrometry (MS)

- Requirement I
  The user shall present the corresponding sample form with formula, name and other required information and samples through posting or in person.

- Requirement II
  If the user is an undergraduate, the user shall submit their sample to their supervisor with approval.

- Requirement III
  The senior researcher shall arrange time and instruments according to their own estimates after receiving the form and sample.

- Requirement IV
  The senior researcher shall run the experiment and charge by the number of used samples and instruments.

- Requirement V
  The user shall get the printed result sheets from senior researcher.

4.4 Solid-State NMR (SS-NMR)

- Requirement I
  The user shall book their experiment with username, machine name, expected time, estimated duration at least 2 weeks in advance.

- Requirement II
  The user and the administrator shall arrange the schedule considering priority and constraints at the meeting.

- Requirement III
  The user shall receive arrangement information from schedule form.

- Requirement IV
  The user shall get extra time for next 2 week arrangements when the cancellation for this week is made.

4.5 X-Ray Crystallography

- Requirement I
  The user shall make appointment in paper with their samples to the experienced academic. First come first serve.

- Requirement II
  The experienced academic shall run the samples on the suitable machine.

- Requirement III
  The experienced academic shall process results on the specific machine with the raw data.

- Requirement IV
  The user shall find their data through the email.

- Requirement V
  Data shall be collected and stored in tapes/CD/DVD.

4.6 Comparison of requirements

Similarities:

- MS and PXRD need the count report as they charge for the survey.
- There is a need for both MS and NOMAD to track the process of experiment.
- PXRD and X-Ray Crystallography arrange running by the booking order. SS-NMR and MS arrange only by the admin decision.
- SS-NMR users have to point out the timeslot as PXRD (Calendar-booking) users do.

Differences:

- There is no fixed schedule for MS, where the estimated time and usage are based on the senior researcher. The length of X-Ray Crystallography running is not predictable, even by academics.
- For the MS, the required information should be relatively changeable.
- SS-NMR does not always charge, depending on the specific experiment, while the X-Ray Crystallography does not charge at all at present.
- The desire of service from users is different. User books PXRD for next 2/7 working days, while SS-NMR is booked for the next 2 weeks, X-Ray Crystallography instruments are arranged immediately and the MS user expects the results before the end of month.
- The running arrangement of MS is distinct from others that the arrangement of running depends on the similarities and classifies of sample.
- The restriction of usage are different. For example, the holder of SS-NMR could be used several times in a day or in a month. At the same time, the holder from PXRD only could be used once a day. The main restriction of SS-NMR user is their personal priority, while the restrictions of PXRD are mainly because of the machine capability.
- The computer skills of technicians are not in the same level for these 4 groups

Chapter 5 Design

This chapter introduces the design of the PXRD booking system from different perspectives, in order to carefully cover all functional and nonfunctional requirements. The first section enumerates use cases with several use case specifications. The second part and third part explain the database design and component design respectively. The representative interface designs are included at the fourth part. At the end, the overall design pattern and structure is briefly explained.

5.1 Use case diagram

The use case diagram graphically represents the communication between stakeholders and systems by translating functional requirements to the use case. A use case embodies a complete sequence of actions, while the whole set of use cases shows the entire measurable values provided by system (Gemino, A., & Parker, D., 2009). Based on functional requirements in Chapter 4.2.1, the user is the only stakeholder/actor in this case. There are two types of users (Standard user and System admin) with different access levels. The access level determines the functionality available to the user, so they are showcased separately in Chapter 5.1.1 and Chapter 5.1.2. The listed use case specifications are chosen as the representatives to explain the documented details of each requirement as specified in Chapter 4.

5.1.1 Standard user

From Chapter 4.2.1, requirements are divided into use cases. Standard user is the actor and all use cases are triggered by user. The identical use case for standard user and admin have the same basic work flow. The alternative and exception flow describes the following steps while scenarios are away from the main scenarios or choices lead to divergent results.

- Figure 3 enumerates all use cases conducted by standard user, which are also applied to admin user. Login use case is specified as the fundamental unit of all use cases. “Add Asset” and “Cancel Asset” exemplify the “Book” and “Cancel” procedures.

Figure 3 Standard user - use case diagram

Identifier	1
Name	Log in
Description	The user logs in with their university credentials
Actors	User
Basic flows	1. User types in username. 2. User types in university email password. 3. User clicks login button. 4. System redirects to the page by the result.
Alternative and exception flows	1. At 3, if user could not be found from approved user list, the result returns as login page.

	2. At 3, if user is not active account, the result returns as login page. 3. At 3, if it is invalid password, the result returns as login page.
Pre-conditions	The user has been already added to the database.
Post-conditions	1. If the password is right, website page jumps to the homepage. The login information of user is recorded to the session attribute. 2. If the user password is not right, website page jumps to the login web page with error information.

Table Use case "Log in" Specification

Identifier	3
Name	Add asset booking
Description	The user makes a reservation for usage of asset-machine (John & Paul).
Actors	User
Basic flows	1. User clicks the name of required machine 2. The website jumps to this machine's booking page. 3. User selects one available date. 4. User selects one provided range. 5. User selects duration from stored choices. 6. User selects unoccupied holder. 7. User clicks Book button. 8. System gets machine restriction settings.
Alternative and exception flows	1. At 4, if the needed range could not be found, the user shall contact the administrator, who will add new program to the machine. 2. At 5, if the needed duration could not be found, the user shall contact the administrator, who will add new program to the machine. 3. At 6, if the needed holder is occupied, user shall contact the administrator or go back to 3, changing date. 4. At 6, if the user is administrator, all holders are available to choose.
Pre-conditions	The user has logged in to his/her account.

Post-conditions	1. If the booking followed booking rule, the new booking will be added to database and web page will reload with success information. 2. If the user is standard user and machine usage or personal daily usage is out of limitation, the web page will reload with fail alert

Table Use case "Add Asset booking" Specification

Identifier	5
Name	Cancel asset booking
Description	User cancels the selected booking record.
Actors	User
Basic flows	1. User checks the booking records. 2. User clicks the cancel button.
Alternative and exception flows	1. At 2, if user is administrator, records will be deleted without restriction. 2. At 2, if user is standard user and current time is after the allowed cancel time, records won't be deleted. 3. At 2, if user is standard user and current time is before the allowed cancel time, records will be deleted.
Pre-conditions	The user has logged in to his/her account.
Post-conditions	Records are updated according to the change and web page is refreshed after operation. If cancellation is made, the success alert will display. Otherwise, the failure alert will display.

Table Use case "Cancel Asset booking" Specification

5.1.2 Admin user

The biggest difference between administrators and standard users is the obligation. The admin user maintains the correctness of the system and makes necessary adjustments. The use cases are categorized by user behaviors and target booking types. First of all, it shows the general management of the system. Then, it introduces use cases in the order of Asset-related, Calendar-related and Onebutton-related.

- Administrative Behaviors

Figure 4 presents a diagram of system management functions. For example, admin could activate/inactivate user account and add/update a new/existing user to manage users. Add/update operation is specified step by step in the use case specification.

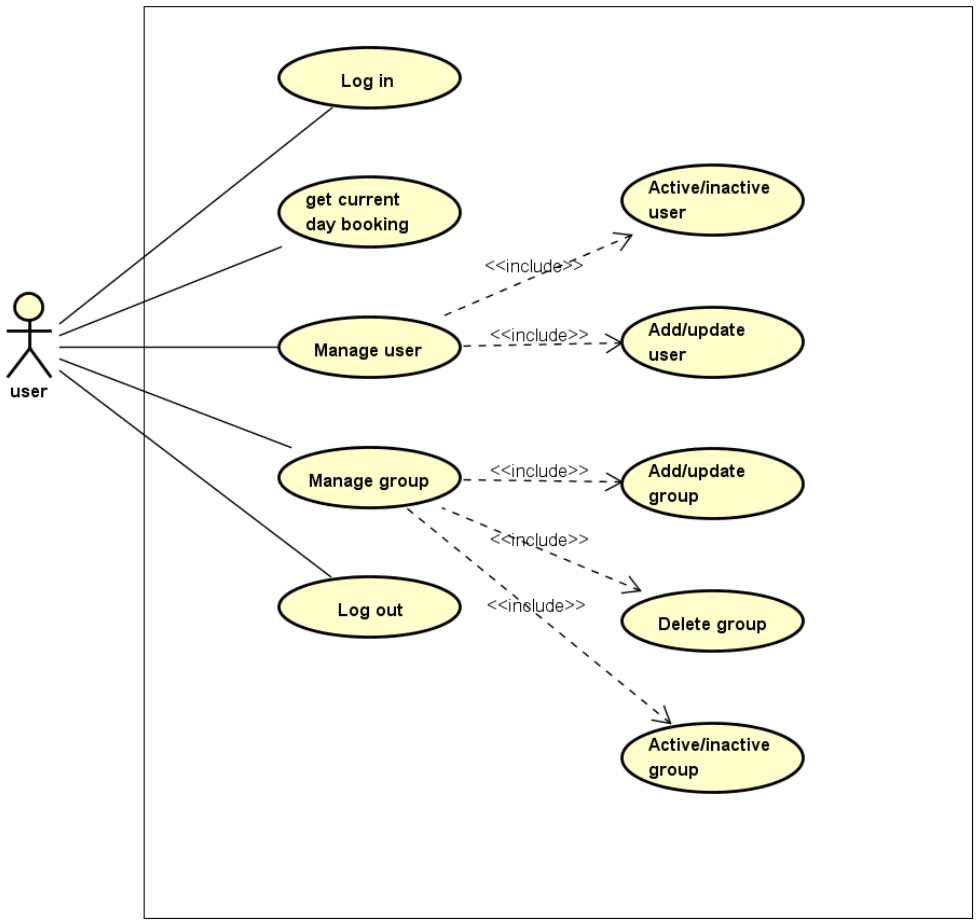


Figure 4 Admin user- system management use case diagram

Identifier	18
Name	Add/update user
Description	Administrator adds new user with personal details. Administrator edits existing user account details.
Actors	User
Basic flows	1. User types in username, real name and email information. 2. User selects associate group name from active group. 3. User sets up user type as system admin or standard user.

	4. User sets up active/inactive attribute. 5. User clicks the save button.
Alternative and exception flows	1. Before 1, user id is generated while user wants to add a new user account. 2. Before 1, user clicks the edit button. Then all details are displayed in every text field. 3. Before 1, user aborts save/update plan, while he/she clicks the clear button, the text fields will be reset. 4. At 1, if there is an invalid input, the page will be reloaded.
Pre-conditions	The user has logged in him-/herself as an administrator account.
Post-conditions	1. If changes are made, records will be updated according to the change and web page is refreshed with the success alert. 2. If exception or errors happened the webpage will be displayed with the failure alert.

Table Use case "Add/update user" Specification

- Asset-related use cases:

  Figure 5 contains all asset-related operations. In addition to the same Book and Cancel actions, the control of relevant equipment and instruments need to be substantiated in an intelligible way, for instance, add/update program.

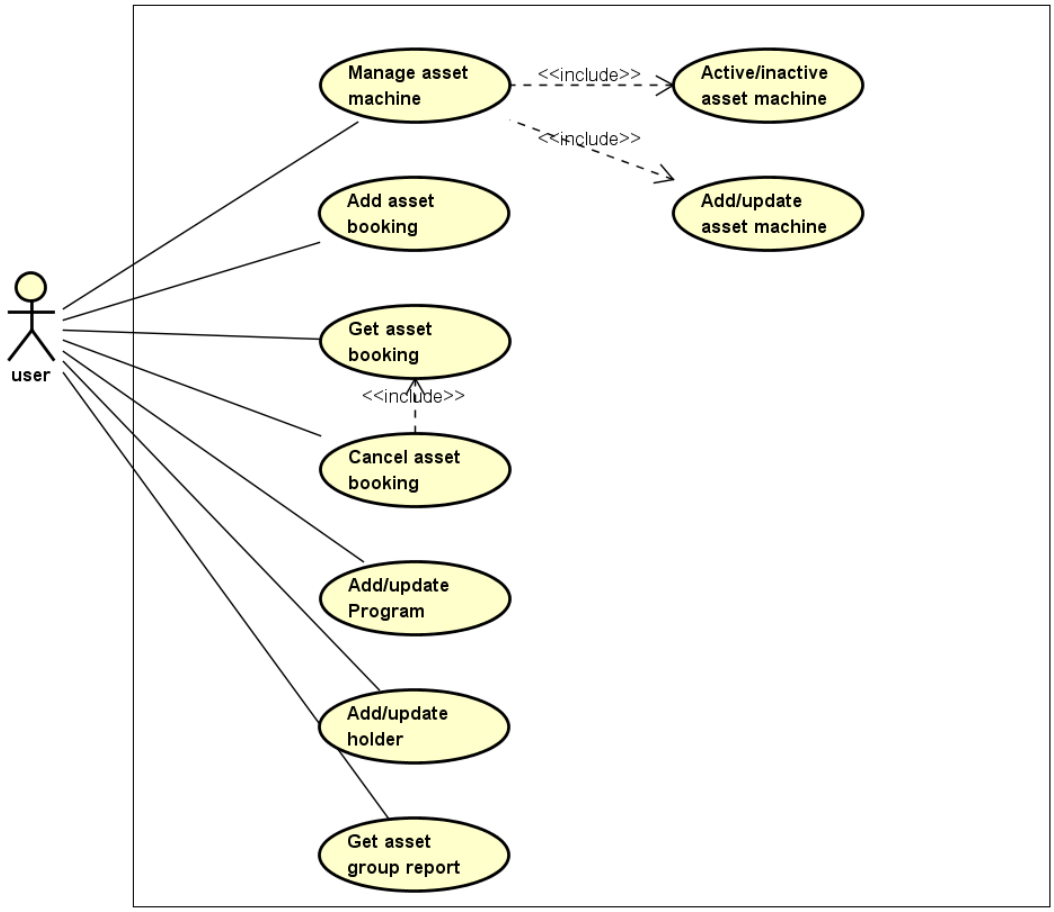


Figure 5 Admin user- Asset-related use case diagram

Identifier	25
Name	Add/update program
Description	User adds new program with duration and range information to the machine. User updates the existing program details.
Actors	User
Basic flows	1. User types in the machine, range and duration information. 2. User clicks save button.
Alternative and exception flows	1. Before 1, program id is generated while user wants to add a new program. 2. Before 1, user clicks the edit button. Then all details are displayed in every text field. 3. Before 1, user aborts save/update plan, while he/she clicks the clear button, the text fields will be reset.

	4. At 1, if machine name does not exist in the system or there is an invalid input, the page will be reloaded.
Pre-conditions	The user has logged in him-/herself as an administrator account.
Post-conditions	1. If changes are made, records will be updated according to the change and web page is refreshed with the success alert. 2. If exception or errors happened the webpage will be displayed with the failure alert.

Table Use case "Add/update program" Specification

- Calendar-related use cases

  Figure 6 presents all calendar-related performance. Similar to the asset-related, manipulations of machine, time-slot and group report (exemplified with specification) are required.

Figure 6 Admin user- Calendar-related use case diagram

Identifier	31
Name	Get calendar group report
Description	User gets calendar machine booking record in selected date range.
Actors	User
Basic flows	1. User selects start date from a standard calendar selector. 2. User selects end date from a standard calendar selector. 3. User clicks search button. 4. Number of booked timeslots and booked overnight timeslots for each machine are shown by group name
Alternative and exception flows	1. At 4, if the group doesn't have booking records during selected period, it will disappear from report result.
Pre-conditions	The user has logged in him-/herself as an administrator account.
Post-conditions	

Table Use case "Get calendar group report" Specification

- Onebutton-related use cases

  The same as asset-related and calendar-related use cases, use case diagram and specification table give concerned performances.

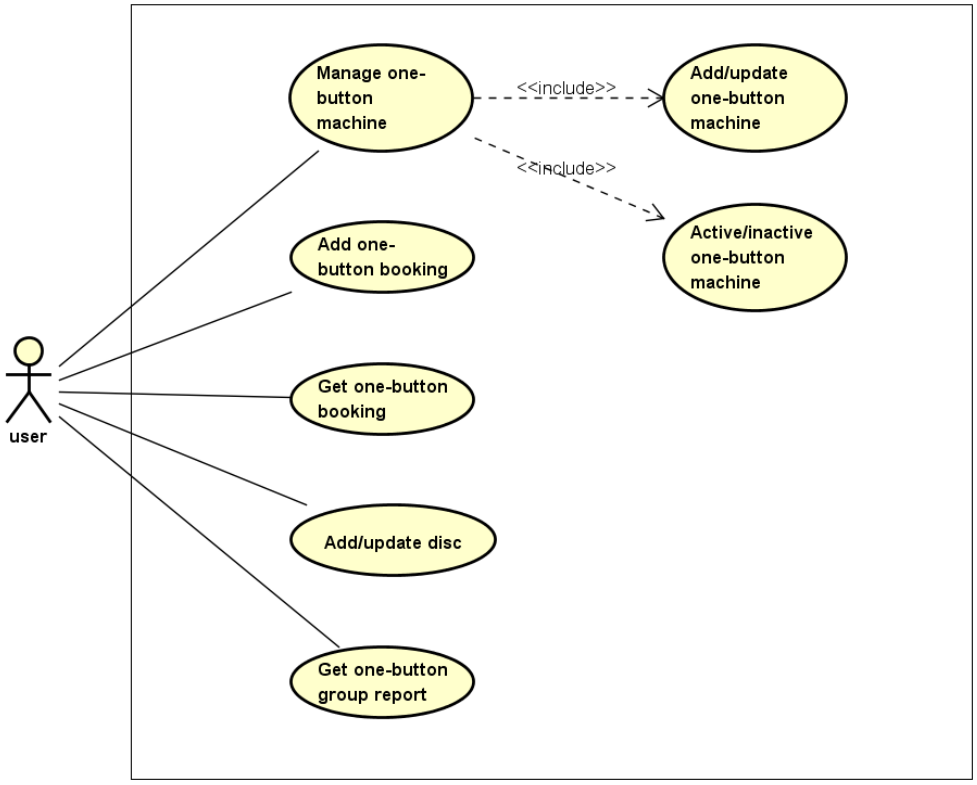


Figure 7 Admin user- Onebutton-related use case diagram

Identifier	33
Name	Active/inactive one-button machine
Description	User changes the availability of machine-activating the inactive machines or inactivating the active machines
Actors	User
Basic flows	1. User finds the target machine. 2. User clicks the inactive button.
Alternative and exception flows	1. At 2, if the machine is currently inactive, user will click the active button.
Pre-conditions	1. The machine exists in the database and has availability attribute. 2. The user has logged in him-/herself as an administrator account.
Post-conditions	1. Database will update availability of machine.

	2. The inactive machine will disappear from side bar menu, while the active machine will reappear to serve user.

Table Use case "Active/inactive one-button machine" Specification

5.2 Database design

Based on the non-functional requirements and the large volume of data, a stable database is an ideal choice for a maintainable, available and reusable system. Due to the varied dimensionality of the data, database shall organize all data in a logical way to allow user searching and updating data. Furthermore, the client (user) and server shall be able to get useful information from database. NOMAD has 46 different tables (see Figure 8) to support its service, where the blue area contains the additional PXRD tables. The PXRD booking system adds its own tables and adapts the ***user***, ***accesslevel*** and ***pigroup*** tables to meet the requirements (see Figure 9).

The database applies to the following normalization:
1NF: database only has atomic values (Chen TX and Liu, SS and Meyer MD, 2007). From Figures 8 and 9, there is no repetition between all underlying domains. For example, each user record from the ***user*** table is equal to a person and each ***calendar_booking*** record from ***calendar_booking*** table represents one booking history. There is no extra id for one person or one booking.

2NF: database is 1NF and each table has its own identity as primary key, where all non-key attribute depends on it (Chen TX and Liu, SS and Meyer MD, 2007). From Figure 8 and 9, it could be easily proved.

3NF: database is 2NF and there is no transitive relationship for non-key attributes (Chen TX and Liu, SS and Meyer MD, 2007). For example, *usedtime* of ***onebutton_booking*** records the machine running time. When a booking is made, the *usedtime* takes the value of current *timetaken* attribute from ***pxrdonebutton_machine*** table searched by *machineid*. While the *timetaken* attribute of each machine is changed, the *usedtime* values of previous bookings are still the same. But the *usedtime* value of new bookings will be different. Therefore, the *usedtime* value only depends on (booking) *id*.

Both Entity Relationship diagrams show the table columns, type, nullable and primary key information. Meanwhile, the foreign keys are represented as dashed lines. The ***asset_booking***, ***calendar_booking*** and ***onebutton_booking*** tables store the booking record with user identity, machine identity, time and their distinct information (e.g. specified range, duration, holder and film). The NMR machines for NOMAD are different from the PXRD machines, where ***nmrmachine*** table is not adaptable to the

PXRD. The ***pxrdasset_machine***, ***pxrdcalendar_machine*** and ***pxrdonebutton_machine*** tables are created with the diffractometer-related information. Each table stores its unique parameters and availability as they have their own restriction rules (e.g. *dailybooking* and *overnightbooking* are the maximum personal daytime and overnight booking times of calendar machine respectively). The ***asset_holder***, ***asset_program***, ***disc*** tables are designed to contain the disc holder and program data. The ***calendar_timeslot*** table stores the start time and end time information as timestamps for each supported time slot.

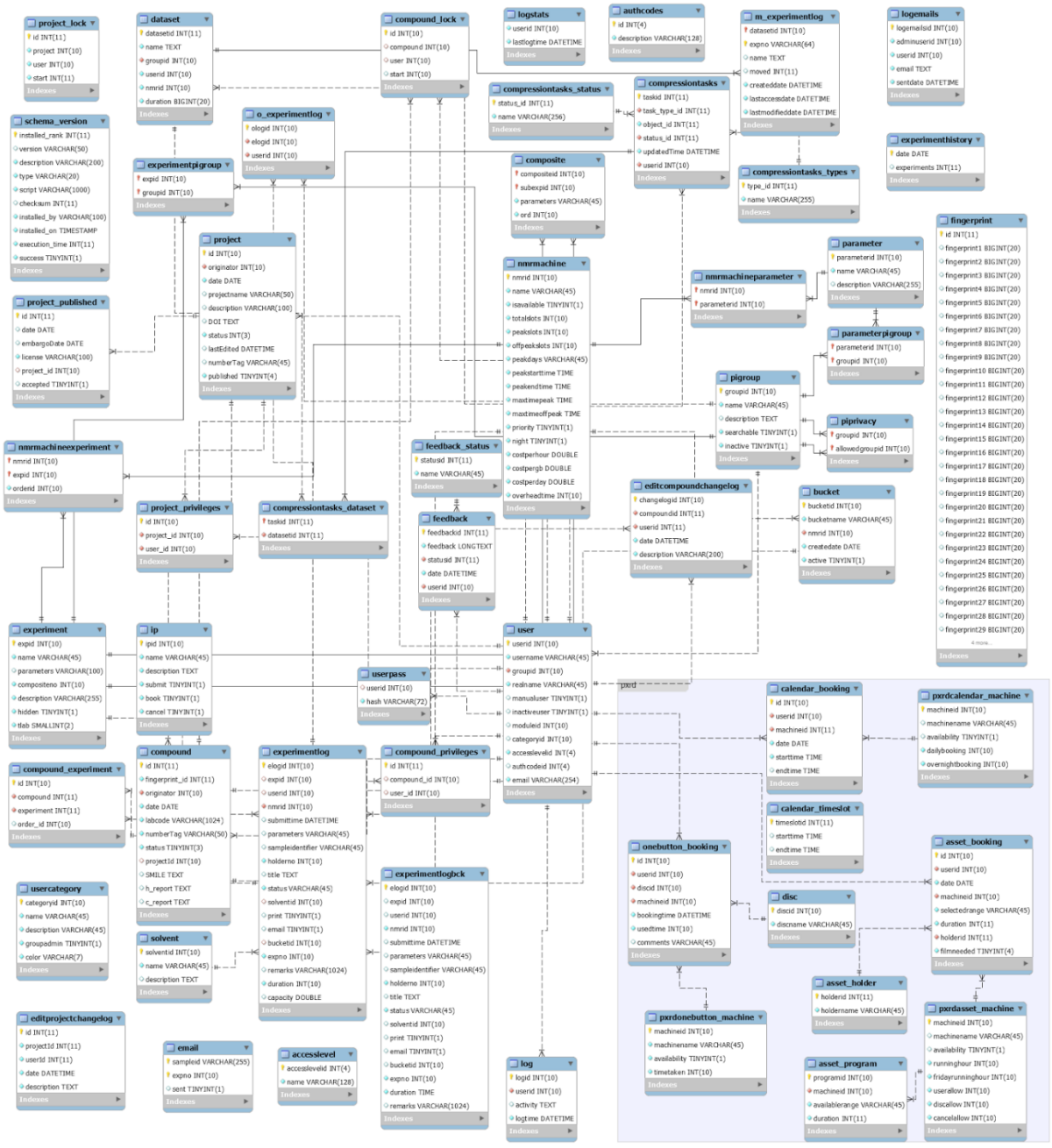


Figure 8 Whole database ER diagram

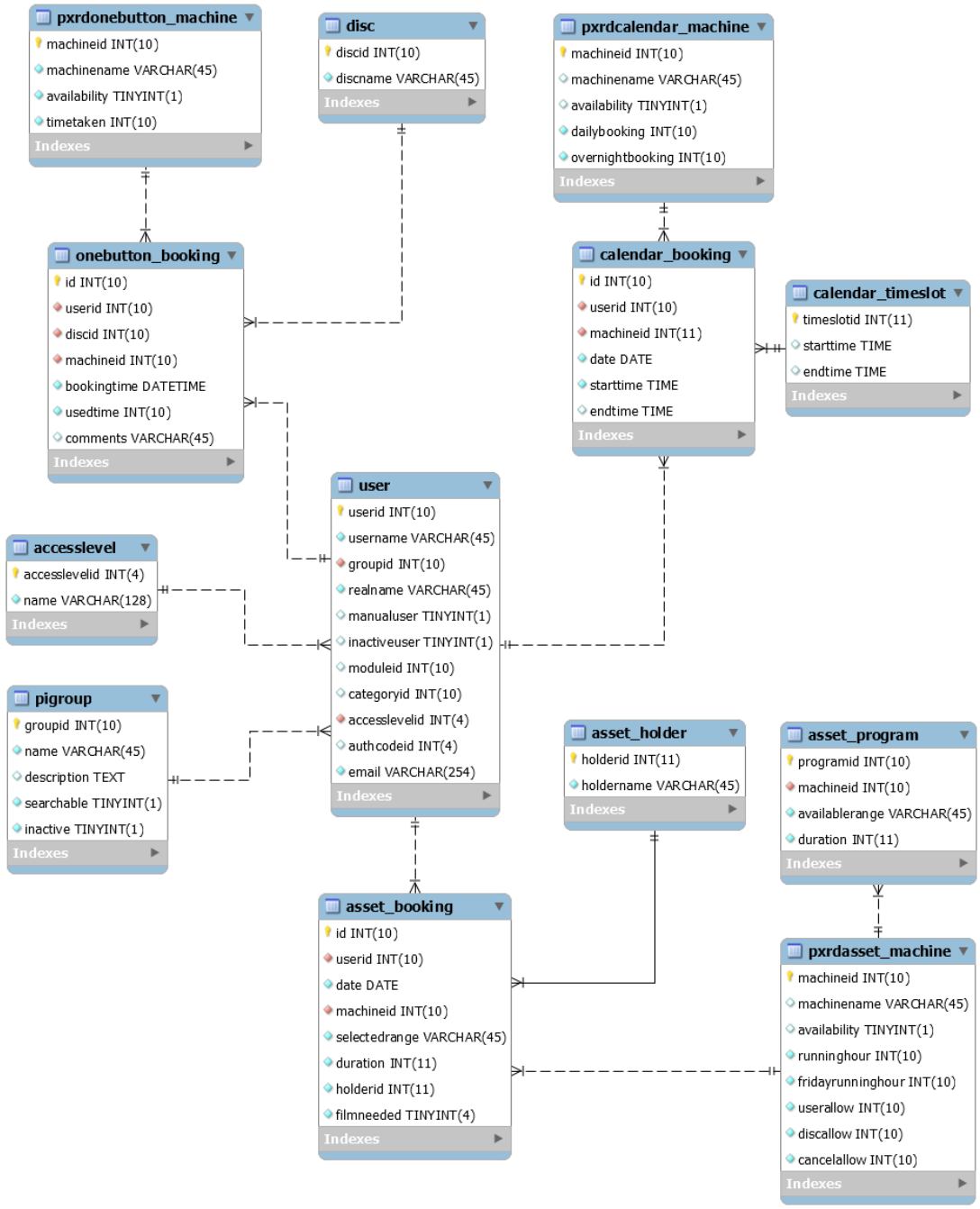


Figure 9 PXRD booking ER diagram

5.3 Component design

It is difficult to apply the requirements of stakeholders to component design, but UML decreases the ambiguity of software processes to some extent (Mahmood, S. & Lai, R, 2009). Therefore, the component diagram (Figure 10) hereby shows the dependencies between each component. In implementation view, a component represents a set of

implementations, which also takes the interfaces from other components (Booch, G., Jacobson, I., & Rumbaugh, J., 1999).

Nomad is a comparatively large system, where it is hard to discuss all detailed designs of its component dependencies. While *nmr-booking* is the main component of the whole project, we'll introduce the design of interfaces and dependencies between *nmr-booking* and the other components. *Nmr-booking* requires the login interface from *nmr-common*, which also provides the needed *UtilityMethods* interface to get formal int, Boolean and date from request. *Nmr-booking* uses the *context initialize interfaces* from *nmr-service* and calendar event manager to get an event object. *Nmr-message* provides the *send email interface* and *nmr-user* provides the *delete group interface* and other useful servlets.

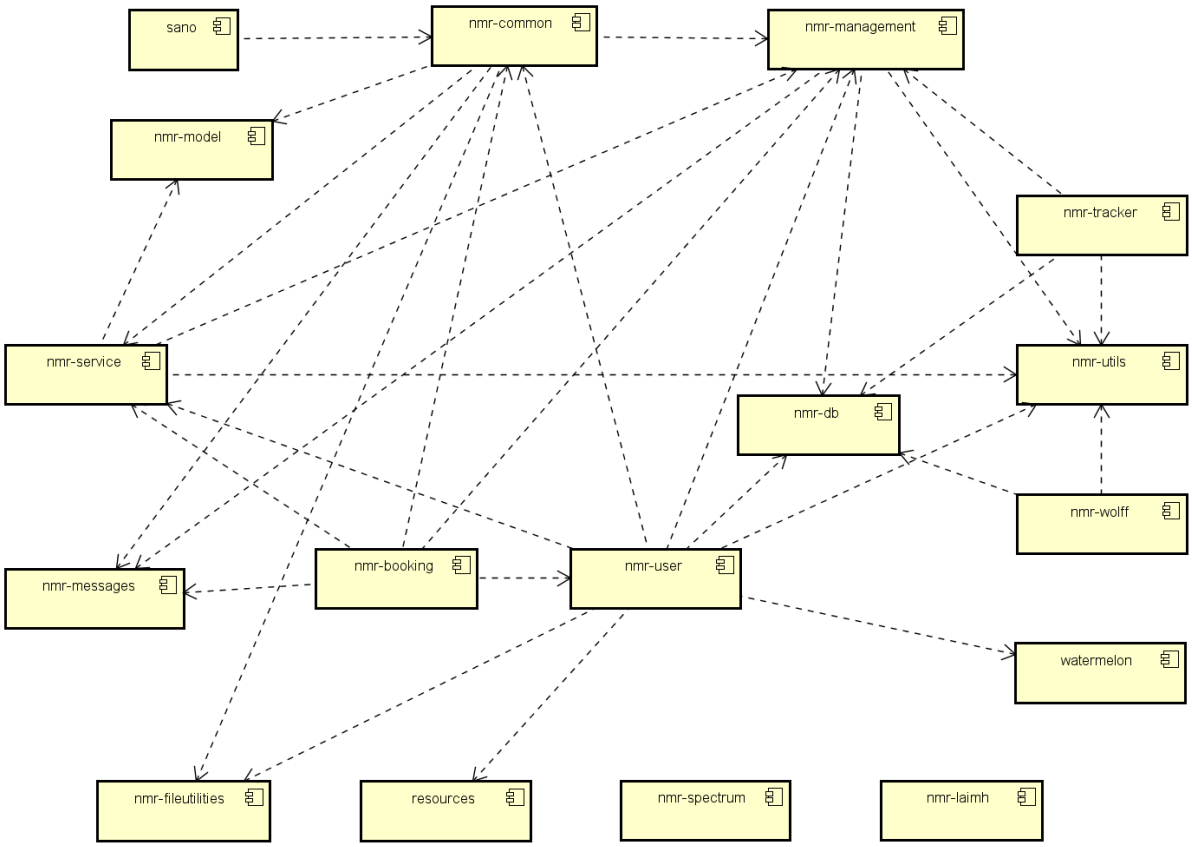


Figure 10 Component diagram

5.4 Interface design

During the requirement gathering process, it was not surprising to find that the computer backgrounds of chemists are diverse. To eliminate misuse and reduce the number of operations, a concise and clear interface is the principle of interface design.

As an external part of NOMAD system, the design style is similar to the original system. Considering the acceptance of operation and functional needs, it is practical to reduce the number of operations as much as possible. Referring to several familiar online booking systems (e.g. www.cheapflights.co.uk, google calendar), the example interface designs are as follows:

1. Asset booking interface: After the user selects the machine name from the side-bar menu, the website redirects to the corresponding machine booking page (see Figure 11). The available booking date according to the booking rule could be chosen from a drop-down list. The same applies to the duration, range and holder. The machine name is populated when the page is loading. This reduces the loss caused by spelling errors. The result list shows the booking records from the last seven days for reference and deletion.

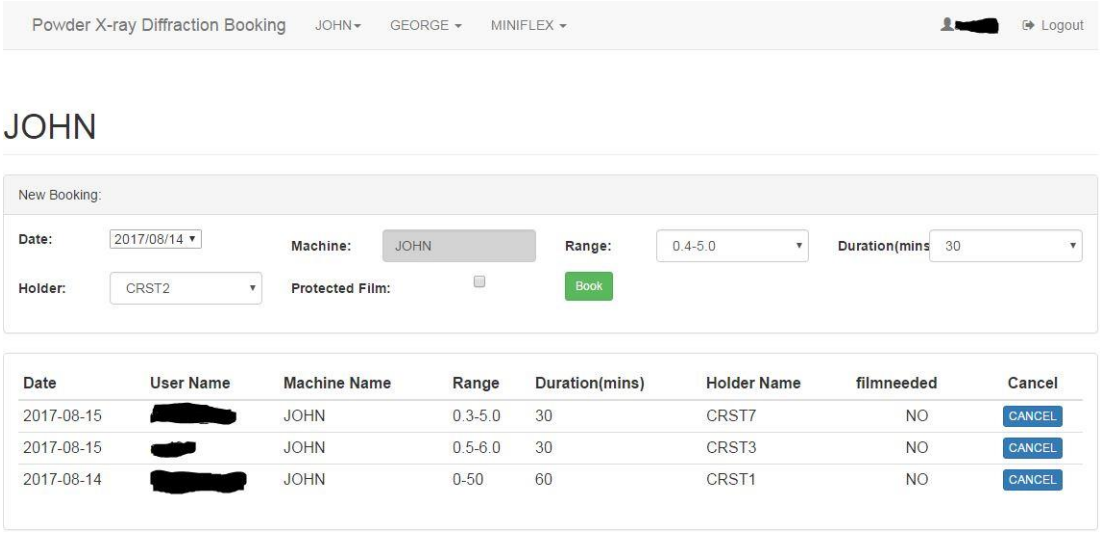


Figure 11 Asset booking interface

2. Calendar booking interface (Figure 12): The plus button and delete button are the only two options for the user to use, both of which are single-click operation. The plus button displays when the timeslots are still available. The user's own bookings are highlighted in red with the delete button to distinguish from others. All user booking records are accessible to all the users. Admin users have all add buttons and delete buttons for all the showed records. When the admin user clicks the plus button, a pop-up window is opened to get user name for booking.

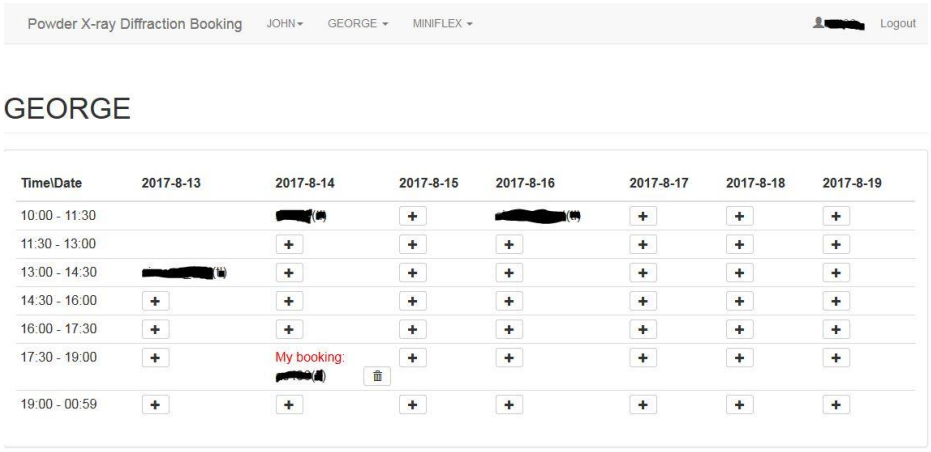


Figure 12 Calendar booking interface

3. Machine management interface (Figure 13): Taking the asset machine as an example, machine management shall allow the user to check with all existing machine settings. The active/inactive operation shall be done by single click operation. When the user makes a change to the machine settings, the page will reload with filled text fields after clicking corresponding edit button. The clear button helps initialize all text fields.

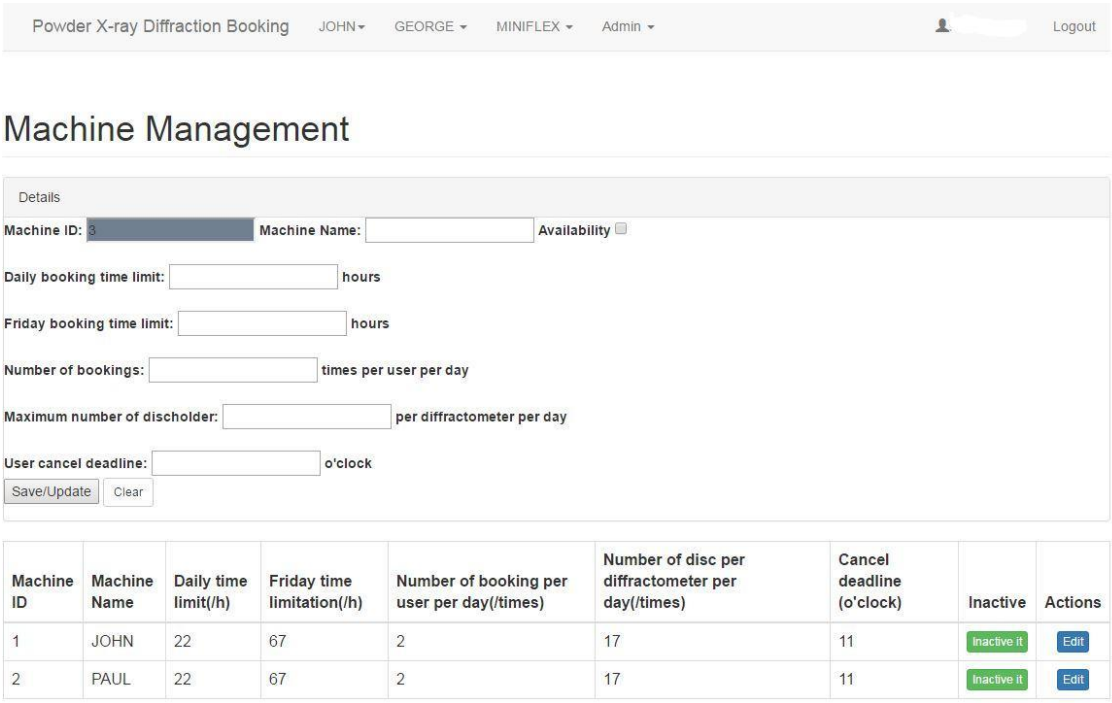


Figure 13 Asset machine management interface

4. Group usage report interface (Figure 14): The side bar menu provides logged user name, logout and all pages' links. Group counts the times of booking and total hours by group name during the selected time period. Report only summarizes the group which has booking records instead of all groups.

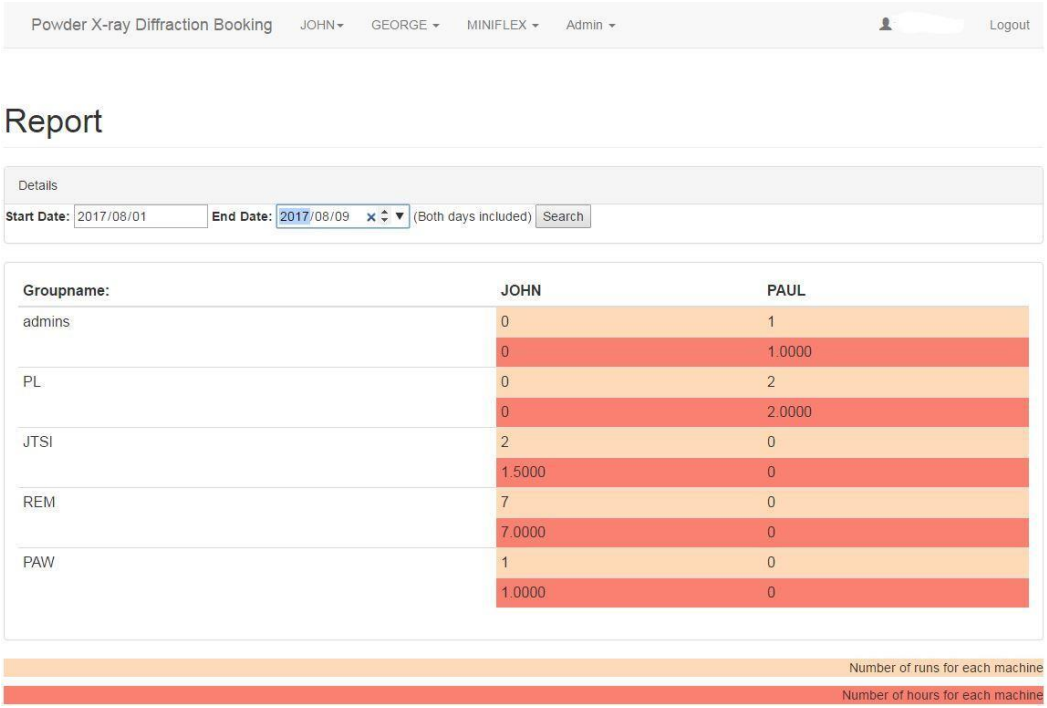


Figure 14 Asset booking group report interface

5.5 Design pattern and structure

To design an interactive system, the Model/View/Controller design pattern is widely used to separate interface from generated data (Leff, A., & Rayfield, J. T., 2001). Model and Controller contain the business logic (Server), while View gives front end interface (Client). JavaEE application model is a good solution to simplify the complexity of development. As it has full support of standardization of Java Servlet API and JSP, the developed web applications have high portability, maintainability and scalability. Following the NOMAD architecture, the JSP/Servlet methods are adapted to implement design by combining the strengths of MVC model and J2EE architecture. JSP files generates all the front-end design, while servlet files provide server support. Both files get information from the database using JDBC. In next chapter, the detailed implementation algorithm and process are introduced with examples.

Chapter 6 Implementation

6.1 Introduction

In this chapter, we discuss the details about how the PXRD booking system was implemented. The architecture, algorithms and alternative solution are discussed in the order of implementation process. Because of the considerable quantity of implemented methods, this section takes several typical algorithms, explained with pseudocode.

The program is version controlled through Mercurial when the progresses have been made. Codes are uploaded at the Kallithea. The as436_testing_branch is used for working version and
as436_trial_and_error branch is used to commit all gradual improvements, even with errors and bugs. The repository can be found at the following address:

as436_testing_branch: http://projectvm03.cs.st-andrews.ac.uk/kallithea/NOMAD-2.0-Development/NOMAD-HEAD/files/as436_testing_branch

as436_trial_and_error: http://projectvm03.cs.st-andrews.ac.uk/kallithea/NOMAD-2.0-Development/NOMAD-HEAD/files/as436_trial_and_error

The in-service system can be found at: http://projectvm03.cs.st-andrews.ac.uk/apps/pxrd. Users need to get proper authorizations to access. Only when users are added to the system by administrator can they use systems.

6.2 Tools, languages and frameworks

6.2.1 UI layer

As an extended service from NOMAD, Intellij is used for the development of the front-end. Bootstrap, HTML, JSTL, JavaScript (with JQuery), Expression language and JAVA are the main languages used in the code.

Bootstrap[12] is an open-source web framework to style the web page design, originally named Twitter Blueprint. In this application, it gives the styling of buttons, navigation menu bar, panel and other layouts. Bootstrap provides extensive HTML, CSS design, as well as JQuery plugin. By importing *bootstrap.min.js* file into JSP pages, all relevant plugin could be used directly.

JavaServer Pages Standard Tag Library (abbr. JSTL)[13] is the JSP based tag library, which is a component of JavaEE web application. It supports SQL and XML taglibs. It retrieves data from SQL queries to the JSP pages. JSTL could be import as taglibs like " <%@ taglib uri="http://java.sun.com/jsp/jstl/..." %>"
Expression language (abbr. EL)[14] started as part of JSTL to dynamically access to data through embedding expression. It allows JSP to use sessionScope, param, paramValues, header and other objects into expression.

6.2.2 Logic Layer

JAVA is one of the most common languages used to build an object-oriented enterprise system. Its "write once, run anywhere" character appeals to 9 million developers developing web applications. Considering the NOMAD architecture coherence and modifiability, in this project, Java is chosen as the developing language for both back-end and front-end.

6.2.3 Persistence Layer

Java Database Connectivity (abbr. JDBC)[15] is a standard API to connect JAVA and database. JDBC are capable to run cross-platform regardless of different databases, realizing accessibility, availability and modifiability consequently. Connection, PreparedStatement and ResultSet are the main methods to implement the most of functional requirements in the servlet.

6.2.4 Tools

[12] http://getbootstrap.com/

[13] https://javaee.github.io/jstl-api/

[14] http://docs.oracle.com/javaee/1.4/tutorial/doc/JSPIntro7.html#wp71019

[15] http://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/

Component	Name	Purpose	Version
Operating System	Windows 7 and Linux	Development Environment	
IDE	Intellij IDEA	Development	2017.1.3 x64
Application server	Apache Tomcat	Server support	8.5.15
Database Client	MySQL Workbench	Database support	6.3CE
Database Server	MySQL (MariaDB)		
Distributed version control	Mercurial	Version control	4.2
Code Review Platform	Kallithea	Peer review	
Web browser	Google Chrome, Mozilla Firefox	Development	

6.3 NOMAD adaptation

- Implementation of Login: A servlet filter intercepts requests to sniff sensitive information. Then the request, including username and password, is sent to the *LoginServlet*, where it compares the username and password from the database and checks whether the group is active. *JSP_pages* (defined in *nmr-service*) are redirected accordingly. Meanwhile, servlet listeners are created to listen to the changes of session.

6.4 One-button booking

- Implementation of activate and inactivate machine:
  There are two machine statuses: active (available) and inactive (unavailable). The essence of changing machine status is to set to the opposite of current status. Therefore, functions $A = 1 \oplus A$ are used to reverse the value.

- Using PreparedStatement to pass input from request to the SQL statement:
  SQL injection is one of the most common hacking methods. It attacks website by malicious SQL statement, including an identical equation (1 = 1), OR condition, updating/deleting SQL sentences and other embedded operations. The PreparedStatement interface from java.sql is the effective protection to prevent SQL injection. *UtilityMethods* from *nmr-common* provides methods to convert string to associated type (int, Boolean, time and date). Then PreparedStatement uses setInt(), setBoolean(), setTime(), setDate() and other methods to assign a value to the SQL sentence. Meanwhile, it converts single quotes to the escape character to avoid the cutoff string value. The PreparedStatement works more efficiently than Statement as well.

- Using JavaScript to check the empty text field:
  Before the request goes to the servlet, the check() function is called to check if there is an empty input. If it exists, the page will return with a failure message before the server connects to the database, which saves bandwidth and strain. Saving unnecessary connection and invalid transfer, the website maintains a high response speed in the server.

6.5 Calendar booking

- Implementation of add-button and delete-button: To minimize the operation steps, the calendar booking interfaces are designed like a real calendar with dates and time slot information (see Figure 15). Each div represents one calendar event object (*CalendarEvent* class from *nmr-service*). The *EventManager* (*nmr-service*) gets event details for each time slot. If there is an existing event from the database, the calendar shows user name and group name. While the user id is the same as the logged user, the delete-button allows the user to delete the booking by given booking id. If there is no event for this div and time is still available to reserve, the add-button allows the user to book the associated time slot with current user identity.

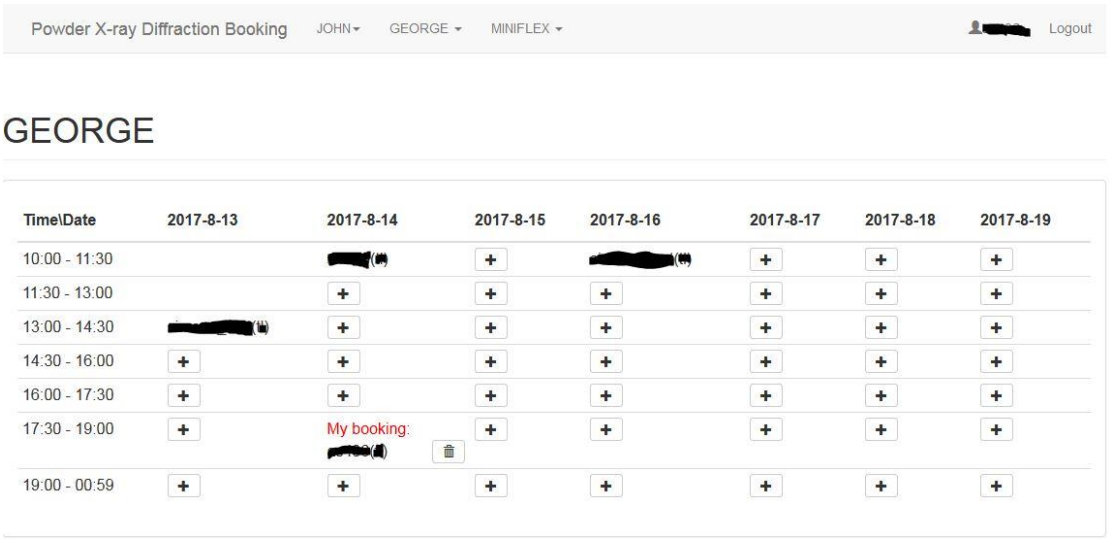


Figure 15 "add-button" and "delete-button"

Algorithm: Assuming: A: User is admin user. B: Booking is mine (user) booking. C: Column represents current day. D: Current time is before the start time. Pa is the no add-button case, while Pa is add-button condition. Pd gives delete-button circumstance.

$$P\bar{a} = \bar{A}C\bar{D} \quad Pa = \overline{\bar{A}C\bar{D}} = A + \bar{C} + D$$

$$Pd = A + \bar{A}B(\bar{C} + D)$$

Alternative solution and comparison: 1) At initial implementation, the Fullcalendar API is used to develop the relevant interface and servlet, providing drags, clicks and other operations. After the first attempt, the timeslot could not be customized, and it is not easy to add an event to the database. Figure 16 gives a weekly scheduler example. 2) At the second implementation stage, a pop-up window is used to add booking details. Changes made after the demo feedback are explained in the next chapter.

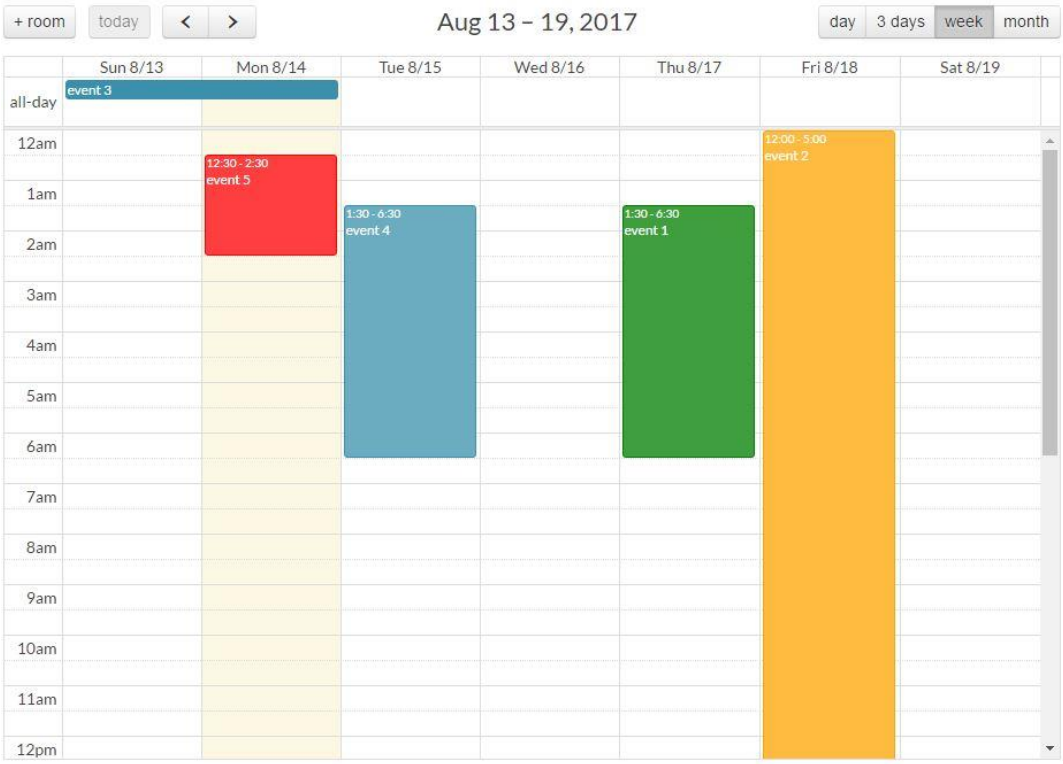


Figure 16 Fullcalendar weekly scheduler

- Implementation of adding calendar booking: When the add-button is clicked, the *addBooking* function is called with several values. The Javascript function first checks whether it is admin to get username. Then by JQuery, the request is sent to the servlet. JQuery is a common-use cross-browser JavaScript library, which provides fast service and massive extendable plugin. JQuery helps to create a reusable and modifiable system. The servlet logic is:

```
Procedure ADDCALENDARBOOKING()
        IF isAdmin THEN {
        addBooking();
        }
    ELSE IF isOvernight THEN {
            IF lessThanOvernightLimitWithin14days THEN {
                    addBooking();
                }
            ELSE {
                    return fail;
                }
                }
            ELSE {
                    IF lessThanDaytimeBookingLimit THEN {
                            addBooking();
                        }
                    ELSE {
                        return fail;
                        }
            }
End procedure

/*There should be less than Maximum allowed overnight times(for example, 1) from
last 7 days and next 7 days. If there were one overnight booking in next 7 days, it should
not allow to book as the frequency would be more than once in continuous 7 days. */

Procedure lessThanOvernightLimitWithin14days(machine A)
    int overnightLimit = A.overnightlimit;
    int count = 0;
    while (overnightRecordHasNext) {
        count += 1;
      }
    IF count < overnightLimit THEN {
        return true;
      }
    ELSE {
        return false;
      }
End procedure
```

- Using alert to show messages: From the non-functional requirement in Chapter 4, it is necessary to give feedback regardless of the success/failure of operation. At servlet part, the redirect URL includes a success parameter to give the result.

However, it is not possible for the user to understand information from URL. Therefore, the success parameter is compared to get right alert (with JQuery) while the page is loading (see Figure 17).

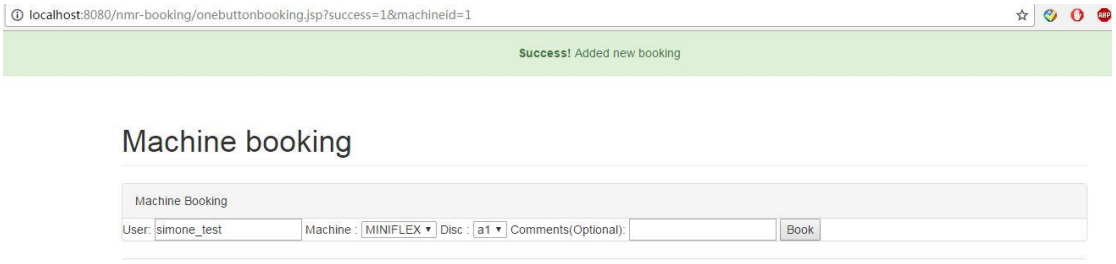


Figure 17 Alert

6.6 Asset-booking

- Implementation of available date:

```
Procedure GETDATES ()
      Date [] =null;
      IF isSaturday THEN {
                  date [0] = Today + 2; date [1] = Today + 3;
            }
      ELSE IF isSunday THEN {
                  date [0] = Today + 1; date [1] = Today + 2;
            }
      ELSE IF isFriday THEN {
            IF beforeCancelTime OR isAdmin THEN {
                  date [0] = Today; date [1] = Today + 3;
            }
            ELSE {
                  date [0] = Today + 3; date [1] = Today + 4;
            }
      }
      ELSE {
            IF beforeCancelTime OR isAdmin THEN {
                  date [0] = Today; date [1] = Today + 1;
            }
            ELSE {
                  date [0] = Today + 1; date [1] = Today + 2;
            }
      }
      RETURN date;
End procedure

Procedure beforeCancelTime()
      Date now;
      Date cancelTime;
      IF now.getTime< cancelTime.getTime THEN {
            return true;
                                                }
      IF now.getTime>=cancelTime.getTime THEN {
            return false;
           }
End Procedure
```

The DatePicker class generates the date array for all possible situations to the jsp. It gives results following the JSTL language, which allows the webpage to present a drop-down selection (see Figure 18).

Figure 18 Date picker

Alternative solution and comparison: 1) The calendar picker is another feasible solution, which only makes two days selectable and others unclickable (see Figure 19). However, in this case, there are only two days available, which does not meet aesthetics with too many irrelevant items. 2) User manually types date in where servlet judges whether it is allowed. In this case, the user could be frustrated to type other information repeatedly, and might not know the available dates. The format of text could also affect the booking result.

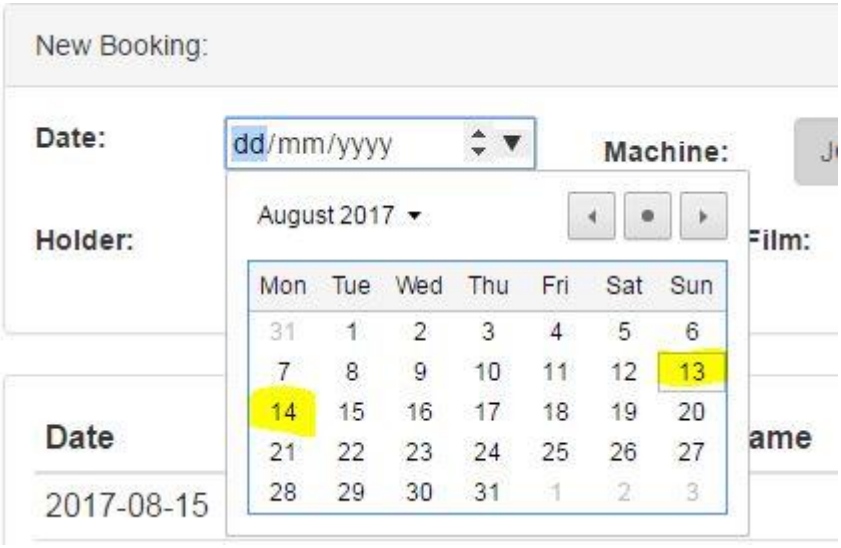


Figure 19 Date picker (calendar way)

- Implementation of selectable range, duration and holder:

The combinations of range and duration are stored at asset_program as optional choices for each machine. Once the range is selected, the matched durations are available to choose (see Figure 20). The same applies to the holder, according to the requirement, only unoccupied holders are available to choose. The algorithm is written in pseudocode.

```
Procedure GETHOLDER()
    usedholder[], unoccupiedholder[] = null;
    usedholder = getUsedHolderByDate (date);
    FOR (int i = 0, i < sizeOfKnownHolder, i++){
        IF(holder[i] NOT IN usedholder) THEN unoccupiedholder[] += holder[i];
    }
    Return unoccupiedholder;
End procedure
```

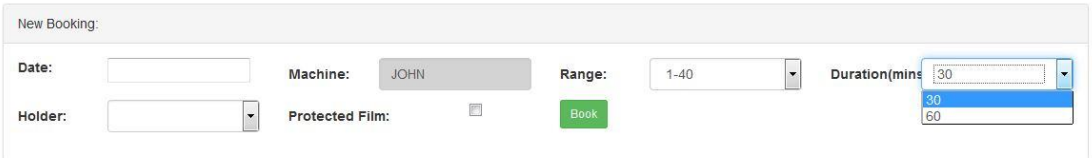


Figure 20 Drop down selection

Alternative solution and comparison: User types range and other information in text box, which user should be responsible for. Comparing to drop-down list, manual input has higher probability of invalid input because of misspelling. Meanwhile, it increases the risk of malicious intrusion, for example, SQL injection.

Chapter 7 Evaluation

7.1 User testing and feedback

According to the iterative development principle, stakeholders provided their input and feedback at interim demonstration sessions at different stages of the project. Stakeholders include Dr Yuri Andreev and common PXRD users (as provided by Dr Andreev with no personal details recorded). Dr Andreev is the manager of the PXRD facility in St Andrews, and also the proposer of this project, referred to as "client" below.

At initial stage, client reviewed the interface designs to tune the proposed functionality with the needs declared during interview. He explained the meaning of the table headings 'Range (2θ)' and 'Time' from the manual booking sheet (Appendix B). The ASCII program file for stored permutation of range and time was provided after the demo, which was not included in the first requirement gathering meeting. The designs of the personal counting report and histogram report were omitted from the perspective of practical use.

At the prototype/demo/status section, the realized functionalities (calendar-related and one-button related) were demonstrated to the client and supervisors. The add button and delete button replaced the dialog box as suggested during demo. We were advised to replace "Asset", "Calendar" and "One-button" with machine names as the category names are defined by the developer.

After the first version of the system was produced, visualization and operations were introduced and questioned one by one. There were 20 new queries in total. Descriptive vocabularies were recommended to make the system more understandable to the admin user. The reports were more informative when they classified data more delineated by timeslot type for calendar and by service time for asset. The overview of the current day was brought up as a new functional requirement. The alternative dates available to the user were shifted from the next 2 days to the next 2 working days.

After the refinement, the PXRD common users performed an unsupervised beta testing of the system to measure the level of acceptance. Because of the nature of critical feedback, the criticism of system is not accordance with the importance and privilege of functionality. In the case of arising conflicts between a user expectation vs the clients (Dr Andreev), the expectation of client prevailed, as his suggestions are more informed, and tailored, with a better understanding of the system and vision for the future. At the last feedback analysis meeting, the client was satisfied with the system design and implementation. Detailed feedback can be found in next section.

7.2 Analysis of Feedback

To obtain feedback, an email was sent to all users in the chemical building by main client Yuri Andreev, containing a link to the questionnaire (which can be found at: https://standrews.eu.qualtrics.com/jfe/form/SV_ekZ8L634H3b6mP3). Candidates information were added by Dr. Andreev in advance. The survey was carried out from 01/08/2017 to 10/08/2017. Out of 35 responses, there were 13 people evaluating the usage of system and 9 people giving suggestions (Feedback report can be found in File *Report.pdf*). 70% and 100% of users found it is easy to login and logout respectively. 81.82% and 100% of users could make and cancel reservations for John and Paul, while 100% and 100% users replied YES to two actions for George and Ringo. There was only 1 user having difficulty to book MINIFLEX. The full feedback report can be found in the appendix.

Requirements for standard users are not always the same as those for the main client. Therefore, a feedback analysis meeting was held to discuss with the developer, client (Dr Yuri Andreev) and supervisors (Tom Kelsey and Shyam Reyal). To identify the future improvements, the user suggestions are listed below.

	User suggestions	Main client analysis	Suggestion/solution
1	“I really like the way you have designed the bookings on George and Ringo! It’s quite simple to use.”		
2	“'Cancel' button for Miniflex”	“There is no need to cancel MINIFLEX booking”	No change needed.
3	“drop-down menu should say ‘disc-mode’ ‘capillary-mode’”	“not an appropriate name, but acceptable”	The menu labels would be replaced.

4	“Please add a small comment box for Paul and John, I would like to keep track of my sample details. This would be very helpful, since currently there is no way to keep track of which sample is in which holder other than writing it on a piece of paper.”	“Don’t need the comment for Paul and John, as users should be responsible for themselves. They should know what they book for. ”	No change needed.
5	“It would be nice if it recognizes that I am part of PAW’s group and offers PAW’s holders as the only one I can select from.”	“Unfortunately, this is not practical for several reasons. Firstly, there are, and will be more in the future that have to be shared. Secondly, there are groups that are “lighter” users and temporary users from other departments and universities. They are sharing holders with no existing-group assignment. And lastly, I think that it is not that difficult to pick an appropriate holder from an alphabetical drop-down list.”	No change needed
6	“Time slots are wrong in the capillary section”		Discussed in Chapter 9.
7	“I still open the old booking system!!!” “Black background not comfortable for the eyes.”		Personal cache and browser issues. No change needed.

Chapter 8 Conclusion and future work

8.1 Conclusion

This project featured an online booking system for the Powder X-ray Diffraction service at the School of Chemistry, following standard software engineering principles. It investigated different booking systems, NOMAD, Agile development, requirements engineering, JavaEE architecture and software testing. The application managed to meet all functional requirements which were documented in Chapter 4. Designs from Chapter 5 were all implemented in the form of Chapter 6. It also gained recognition of its functionality as stated in the feedback

The PXRD booking system allows the user to book experiments with all five PXRD machines with embedded booking rules. The cancellation and review of booking alleviate the burden on researchers. It helps administrators to digitize their management work and raises the financial rewards.

This project doesn't only give me the industrial experience, but also boosts my confidence. It allows me to combine practice and the knowledge from my courses (Software Engineering Principle and Practise, Object-Oriented Modelling, Design and Programming, Computer Security and Critical Systems Engineering). I felt a great sense of accomplishment when the system functioned properly after deployment and when the client and users expressed their approval. I believe that the design, development and maintenance of a live client project is the best way of understanding software engineering at its best.

8.2 Critical appraisal and evaluation of objectives

The functional requirements from Chapter 4 are all satisfied under the design from Chapter 5. It implicitly meets the non-functional requirements. The website has been functioning properly from different web browsers and operating systems. The objectives matrix shows the overall accomplishment of objectives (FA-Fully Achieved, PA-Partly Achieved, NA-Not Achieved).

Objectives	Status	Statement

Primary objectives		
1. General requirement gathering and elicitation from multiple departments in chemistry, including Powder X-Ray diffraction	FA	Finished during 15/05/2017 - 05/06/2017
2. The generic model design and identification of differences between different machineries	FA	Finished during 15/05/2017 - 08/06/2017
3. Software implementation for calendar and one-button booking modes in PXRD	FA	Finished during 13/06/2017 – 21/07/2017
4. Client's and users' evaluation of PXRD booking system	FA	Finished gradually in different stages of development
Secondary Objectives		
1. Software implementation for asset-booking mode	FA	Finished during 22/07/2017 – 31/07/2017
2. Adapting calendar and one-button booking modes to other machineries	PA	One-button applies to X-Ray Crystallography and calendar applies to Solid State NMR in theory.
3. Evaluation of booking system for other machineries	NA	Could be finished with future real cases.
Tertiary Objectives		
1. UX/UI analysis and implementation	FA	Finished gradually in different stages of development
2. Investigating booking machinery using mobile message bots	NA	Could be finished in the future work

8.3 Known issues & Limitations

- Tomcat time zone problem:
  Affected by Daylight Saving time, Tomcat is showing one hour different from the real time. For example, the calendar timeslot and one-button records show one hour later than the real timeslot (Figure 21).This is a problem with configuration in the development server and would be automatically resolved when the system is deployed in the IT services VM cluster.

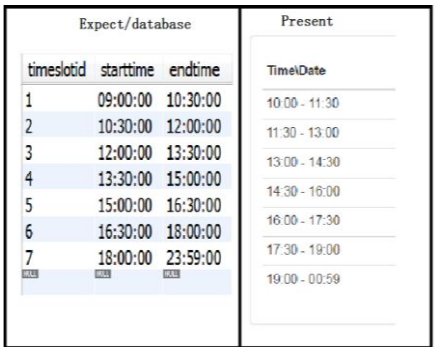


Figure 21 Tomcat time zone problem

- Log in problem:
  When the user enters their identity and password, the system has already logged with userid in the Session. Instead of redirecting the home page, the login page is reloaded again. After the second time, the homepage shows up. This problem only happened once a day, which may be caused by the context path.

8.3 Future work

Based on the user experiences, secondary and tertiary objectives, there are some future extensions, which could be improved by the NOMAD team.

- Sending email notifications

  The admin should be able to send a mass notification to entire groups, or department. After the final feedback meeting, the client brought up the new requirements inspired by the NOMAD functionalities. With the time limitation of this project, this function would be finished by the NOMAD team.

- Adapting Print Service API

  The admin shall be able to print out the current day booking records for easy reference and display. The instrument manager brought up the new requirement because of his personal computer skills. The JAVA Print Service API could help

implement this function. The NOMAD team will on this requirement after completion of this project.

- Applying to other techniques

  With time restriction, it is not feasible to conduct the experiments on other machineries/techniques with real cases. The methodology and requirements have been infused into the project design. The system ought to be adaptable by new techniques and if other developers were willing to attempt this, its reusability would be enhanced.

- Investigating booking machinery using mobile message bots

  With the rise of smartphones, mobile message bots would be a better choice than just email notification. It also could combine the booking with a mobile calendar reminder. By the requirement and revenue of the stakeholders, the developer cannot investigate this further. But, if the system would be used on large-scale enterprises in the future, this function would be an interesting feature.

Bibliography

Booch, G. (1986). Object-oriented development. *IEEE Transactions on Software Engineering*, SE-12(2), pp.211-221.

Booch, G., Jacobson, I., & Rumbaugh, J. (1999). The Unified Modeling Language Reference Manual. https://doi.org/10.1017/CBO9781107415324.004

Chen, T., Liu, S., & Meyer, M. (2007). An introduction to functional independency in relational database normalization. Proceedings of the 45th, 221–225. https://doi.org/10.1145/1233341.1233381

De Vora, A., Auer, M. E., & Grout, I. (2007). A general framework and booking system for online laboratories based on open source technologies. In Innovations in E-learning, Instruction Technology, Assessment, and Engineering Education (pp. 45–49). https://doi.org/10.1007/978-1-4020-6262-9_8

Ferreira, J. M. M., & Cardoso, A. M. (2005). A Moodle Extension to Book Online Labs. International Journal of Online Engineering, 1(2), 1–7.

Gemino, A., & Parker, D. (2009). Use Case Diagrams in Support of Use Case Modeling: Deriving Understanding from the Picture. Journal of Database Management, 20(1), 1–24. https://doi.org/10.4018/jdm.2009010101

Hinsen, K., Läufer, K. and Thiruvathukal, G. (2009). Essential Tools: Version Control Systems. *Computing in Science & Engineering*, 11(6), pp.84-91.

Jalote, P. (2005). *An Integrated Approach to Software Engineering*. 3rd ed. India: Narosa Publishing House.

Knight, J. (1972). A case study: Airlines reservations systems. Proceedings of the IEEE, 60(11), pp.1423-1431.

Leff, A., & Rayfield, J. T. (2001). Web-application development using the Model/View/Controller design pattern. Proceedings - 5th IEEE International Enterprise Distributed Object Computing Conference, 2001–Janua(January), 118–127. https://doi.org/10.1109/EDOC.2001.950428

Li, Y., Esche, S. K., & Chassapis, C. (2008). A scheduling system for shared online laboratory resources. In Proceedings - Frontiers in Education Conference, FIE. https://doi.org/10.1109/FIE.2008.4720253

Mahmood, S., & Lai, R. (2009). RE-UML: An extension to uml for specifying

component-based software system. Proceedings of the Australian Software Engineering Conference, ASWEC, 220–228. https://doi.org/10.1109/ASWEC.2009.28

McTavish, C., & Sankaranarayanan, S. (2010). Intelligent agent based hotel search & booking system. In 2010 IEEE International Conference on Electro/Information Technology, EIT2010. https://doi.org/10.1109/EIT.2010.5612121

Pandey, D., Suman, U. and Ramani, A. (2010). An Effective Requirement Engineering Process Model for Software Development and Requirements Management. *2010 International Conference on Advances in Recent Technologies in Communication and Computing*, pp.287-291.

Sommerville, I. (2016). *Software engineering*. 10th ed. Harlow: Pearson Education.

Teuber, C., & Forbrig, P. (2004). Different types of patterns for online-booking systems. Proceedings of the 3rd Annual Conference on Task Models and Diagrams - TAMODIA '04, 91. https://doi.org/10.1145/1045446.1045464

W., F., S., D., T., G. and P., X. (2012). Design and realization of laboratory information magement system based on J2ME-J2EE. 12(6), pp.24-27.

Appendix A – Ethics

UNIVERSITY OF ST ANDREWS
TEACHING AND RESEARCH ETHICS COMMITTEE (UTREC)
SCHOOL OF COMPUTER SCIENCE
ARTIFACT EVALUATION FORM

Title of project

Scientific Data Management System for Powder X-Ray Diffraction

Name of researcher(s)

Anke Shi

Name of supervisor

Shyam Reyal; Tom Kelsey

Self audit has been conducted **YES** ☒ **NO** ☐

This project is covered by the ethical application CS12476

Signature Student or Researcher

Anke Shi

Print Name

ANKE SHI

Date

22/05/2017

Signature Lead Researcher or Supervisor

Print Name

TOM KELSEY

Date

22/5/2017

Computer Science Preliminary Ethics Self-Assessment Form

Research with human subjects

Does your research involve human subjects or have potential adverse consequences for human welfare and wellbeing?

YES ☒ **NO** ☐

If YES, full ethics review required
For example:
Will you be surveying, observing or interviewing human subjects?
Will you be analysing secondary data that could significantly affect human subjects?
Does your research have the potential to have a significant negative effect on people in the study area?

Potential physical or psychological harm, discomfort or stress

Are there any foreseeable risks to the researcher, or to any participants in this research?

YES ☐ **NO** ☒

If YES, full ethics review required
For example:
Is there any potential that there could be physical harm for anyone involved in the research?
Is there any potential for psychological harm, discomfort or stress for anyone involved in the research?

Conflicts of interest

Do any conflicts of interest arise?

YES ☐ **NO** ☒

If YES, full ethics review required
For example:
Might research objectivity be compromised by sponsorship?
Might any issues of intellectual property or roles in research be raised?

Funding

Is your research funded externally?

YES ☐ **NO** ☒

If YES, does the funder appear on the 'currently automatically approved' list on the UTREC website?

YES ☐ **NO** ☐

If NO, you will need to submit a Funding Approval Application as per instructions on the UTREC website.

Research with animals

Does your research involve the use of living animals?

YES ☐ **NO** ☒

If YES, your proposal must be referred to the University's Animal Welfare and Ethics Committee (AWEC)

University Teaching and Research Ethics Committee (UTREC) pages
http://www.st-andrews.ac.uk/utrec/

UNIVERSITY OF ST ANDREWS
TEACHING AND RESEARCH ETHICS COMMITTEE (UTREC)
SCHOOL OF COMPUTER SCIENCE
PRELIMINARY ETHICS SELF-ASSESSMENT FORM

This Preliminary Ethics Self-Assessment Form is to be conducted by the researcher, and completed in conjunction with the Guidelines for Ethical Research Practice. All staff and students of the School of Computer Science must complete it prior to commencing research.

This Form will act as a formal record of your ethical considerations.
Tick one box

- ☐ **Staff Project**
- ☒ **Postgraduate Project**
- ☐ **Undergraduate Project**

Title of project

Scientific Data Management System for Powder X-Ray Diffraction

Name of researcher(s)

Anke Shi

Name of supervisor (for student research)

Shyam Reyal; Tom Kelsey

OVERALL ASSESSMENT (to be signed after questions, overleaf, have been completed)

Self audit has been conducted **YES** ☒ **NO** ☐

There are no ethical issues raised by this project

Signature Student or Researcher

Anke Shi

Print Name

ANKE SHI

Date

22/05/2017

Signature Lead Researcher or Supervisor

Print Name

TOM KELSEY

Date

22/5/2017

This form must be date stamped and held in the files of the Lead Researcher or Supervisor. If fieldwork is required, a copy must also be lodged with appropriate Risk Assessment forms. The School Ethics Committee will be responsible for monitoring assessments.

Appendix B - Raw requirements from each service

- Asset machine booking sheet:

Total time not to exceed 22 Hours

John Stack Queue Sheet 1 Date:11/05/17

Enter your disc number, research group and range below and your sample will be dealt with accordingly.

	Disc name	Research Group And User Name	Range (2θ)	Time
A1				
A2				
A3				
A4				
A5				
A6				
A7				
A8				
A9				
A10				
A11				
A12				
A13				
A14				
A15				
B				
C				

Monday-Thursday Total 22hrs

Samples to be presented for loading before 12:00.

Samples will be unloaded the following morning at @11:30

Friday runs can be longer, up to a total of 67 Hours

- Calendar machine booking sheet

STOE BOOKING SHEET GEORGE 15/05/17

DESIGNATED USERS ONLY **CAPILLARY MODE ONLY**

ALL USERS BOOK 1.5 HOUR SLOTS

Please refer to 'The New Rules' before booking time! |**User Name and Group**

	9.00 - 10.30	10.30 - 12.00	12.00 - 1.30
Mon			
Tue			
Wed			
Thu			
Fri			
Sat			
Sun			

	1.30 - 3.00	3.00 - 4.30	4.30-6.00	Over night run
Mon				
Tue				
Wed				
Thu				
Fri				
Sat				
Sun				

- SS booking sheet:

Not From Ashbr
group

University of St Andrews

Sample Submission for the Solid-State NMR Service

(academic research samples only)

Please contact Daniel (dmd7@st-andrews.ac.uk) to discuss the suitability of solid-state NMR for your work before submitting samples. Spectrometer usage is charged at the rate of £2/sample for "short" experiments (^{1}H, ^{19}F, ^{23}Na, ^{27}Al, ^{31}P, *etc.*) and £5/sample for "long" experiments (^{13}C, ^{29}Si, ^{77}Se, shorter 2D spectra, *etc.*). For very long experiments (^{15}N, ^{25}Mg, ^{125}Te, *etc.*, as well as longer 2D spectra and non-standard experiments), spectrometer usage will be charged at the rate of £50 per day.

Please note that you will only be able to access your data once you have collected your samples.

Name _____ **Group** _____

Email ID _____ **Date** _____

Sample details (sample code, structure, description, *etc.*)

☐ This sample is safe (hazard class 1) and stable to air/moisture. If not, any toxicity/special handling information **MUST** be included here.

Experiments required

Discuss this with Daniel before submitting the samples if you are not sure.

Grant to charge _____

Supervisor's signature _____

- MS booking sheet:

2 YEAR

University of St Andrews
School of Chemistry

Mass Spectrometry Service

- Chemistry only

Name of Researcher – *

E-mail – *

Supervisor – *

Lab Book Code Number – * ID FOR THE SAMPLE

COSHH Assessment Code Number – OPTIONAL

admin ↓ Aitken Alan (David) VIA CHARM. COSHH (Nouchali)

Possible Structural Formula – DRAWN

Probable Molecular Formula – Text

Nominal Mass – ~~Text~~ Number

Melting Point in °C* – Number

* required for EI samples

CONFIRMABLE

Ionisation Method Requested –

Electron Impact – ✓

APCI – ☐

Chemical Ionisation – ☐

~~GCMS – ☐~~

Electrospray – ✓ → sample soluble in . TEXT -

Resolution Requested –

Low Resolution – ✓

High Resolution – ✓ on *m/z* NUMBER

Additional Requirements / Information (continue on reverse of form if necessary) –

Signature of Researcher – *

Authorised by – *

Date Submitted – *

Grant Code to Charge – *

For service use only –