FrAnK: A Web Application for the Annotation of Mass Spectral Peaks Using Fragmentation Spectra

Scott J. Greig

School of Computing Science

Sir Alwyn Williams Building

University of Glasgow

G12 8RZ

A dissertation presented in part fulfillment of the requirements of the Degree of Master of Science at the University of Glasgow

07/09/15

Abstract

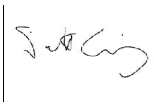
In the field of Metabolomics, biological samples are routinely analysed using Mass Spectroscopy (MS) techniques with the aim of quantifying and identifying the constituent metabolites. The University of Glasgow Metabolomics Facility has developed a proprietary web-based application (PiMP) for the analysis of mass spectral data generated by the research staff using this technique. However, PiMP does not support the extraction and storage of fragmentation patterns, which are analogous to a structural fingerprint, thereby limiting the veracity of metabolite identification.

To provide supporting evidence for putative metabolite identifications, a web-based application to maintain the lineage of MS peaks and utilise their fragmentation spectra in the retrieval of candidate annotations was developed. The Fragment Annotation Kit (FrAnK) is a Django-based application which implements Celery to facilitate the asynchronous processing of MS data sets. Interfacing with R, peak data is derived and stored from mzXML source files using scripts tailored to the experimental protocol. The hierarchical fragmentation spectra are utilized in the retrieval of candidate annotations via spectral reference libraries. In the form of a SOAP request, fragmentation spectra may be submitted for analysis using the MassBank Web API. Alternatively, the libraries of NIST14 are queried via the Windows-based MS PepSearch software, supported within a Linux-like environment through Wine. As each spectral reference library may generate numerous candidate annotations for a given peak, the user may select a preferred candidate annotation, proposing putative metabolite identification, from those annotations retrieved.

While limitations have been identified in the FrAnK application, the development provides a framework to support the integration of novel software and algorithms for the identification of metabolites.

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic form.

Name: Scott Greig Signature: 

Acknowledgements

I would like to thank both the project supervisor Dr Simon Rogers and the PhD researcher Joe Wandy, who served as the secondary supervisor, for both their technical support and guidance throughout all phases of the project. In addition, I would like to thank all the members of staff at the Metabolomics Facility of Glasgow Polyomics who kindly offered their support and the imparting of invaluable technical and domain knowledge. Of note were Yoann Gloaguen, Dr Ronan Daly, Dr Stefan Weidt and Dr Justin Van Der Hooft. In addition, I would like to thank Dr Karl Burgess, the Head of Metabolomics, for assisting in the evaluation of prototypes, providing guidance on the direction of development and the evaluation of the final product.

In addition, I would like to recognize the contributions of Dr Simon Rogers, Joe Wandy, Dr Karl Burgess and Tony Lawson for providing code which has been integrated and acknowledged in the comments documenting the code. Dr Rogers provided additional 'Annotation Tools', namely the 'Precursor Mass Filter' and the 'Network Sampler', which have been implemented in FrAnK. Joe Wandy and Tony Lawson provided the 'frankMSnPeakMatrix.R' and 'frankXcmsSetFragments.R' scripts, respectively, located in the 'Frank_R' folder of the application. In addition, the 'gcmsGeneratePeakList.R' script within the same folder is an adaptation of an R script which was provided by Dr Karl Burgess.

Contents

Chapter 1 Introduction

Historically, life sciences research has been predicated upon highly targeted experimental approaches to investigate specific molecules of interest within biological systems. Just as the three blind men were unable to reach a consensus upon the shape of the elephant, life scientists have acknowledged that investigating the complexities of biological systems in isolation is ineffectual and the adoption of more holistic, untargeted approaches are essential. Potential biomarkers of pathological states, novel therapeutic-targets and diagnostic biomarkers may go unidentified (Courant *et al.*, 2014). Since the inception of genomics and transcriptomics in the 1980s, there has been increasing collaboration between life-sciences researchers and computing scientists to meet the challenges of new, holistic approaches within the biological "*omics*" fields.

In contrast to related *–omics* fields such as genomics, transcriptomics and proteomics, to date the emerging field of metabolomics has received relatively limited attention from computing scientists. In part, this has been due to the relative infancy of the field and the challenge of bridging the gap between life sciences domain knowledge and technical proficiency (Smith *et al.*, 2014). As such, attempts to develop software to identify small molecular metabolites have been met with limited success to date. However, there is a concerted effort within the field to improve the computational framework supporting this burgeoning scientific discipline. In collaboration with the Metabolomics Facility at Glasgow Polyomics, the project aims to develop a web application to aid researchers in the identification of small biological metabolites using fragmentation patterns generated using mass spectrometry. However, the field of metabolomics, including the fundamentals of mass spectrometry, will initially be reviewed to familiarise the reader with an overview of the necessary domain knowledge.

1.1 Background

1.1.1 Metabolomics

Since its introduction approximately 15 years ago, Metabolomics has been an emerging discipline focused upon the high-throughput quantification and identification of small molecular metabolites (typically 50-1500 Da) which are synthesized by the metabolic pathways of biological systems (Courant *et al.*, 2014; Smith *et al.*, 2014). Cellular physiology can be considered from three distinct levels – gene expression and transcription (genomics and transcriptomics; *the blueprint of cellular processes*), protein expression and state (proteomics; *the machinery which drives cellular processes*) and the small molecules which serve as either substrates or products of cellular pathways (*the net effect of cellular processes*). The latter of which is the concern of the field of Metabolomics, which reflects the phenotype or state of the metabolic processes of a biological system.

In targeted metabolomics investigations, a selective subset of chemically-related metabolites belonging to a specific cellular pathway of interest will be

investigated (Courant *et al.*, 2014). The advantage of such targeted approaches is they are hypothesis-driven and, as such, the cellular pathway or metabolites of interest are typically defined in advance. This allows for relatively simple quantification and identification, however, the interpretation of such data is limited in scope due to the selectivity of the experimental design. An alternative is to adopt a 'metabolic fingerprinting' approach, allowing for the statistical analysis of the wider metabolome across experimental factors (Courant *et al.*, 2014). This experimental approach is untargeted, and as such is open to new findings. However, due to the lack of selectivity for a specific cellular pathway or metabolite, the challenge remains to identify the metabolites of interest.

Defined as "all the metabolites within an organism", the metabolome encompasses both endogenous (such as amino acids, lipids, organic acids and bases) and exogenous (such as xenobiotics) metabolites (Courant *et al.*, 2014; Glish and Vachet, 2003). Furthermore, the metabolites which comprise the metabolome of a biological system may vary in their chemical and physical properties, concentration and distribution within an organism. Despite the associated challenges, improving the veracity of metabolite identification will not only further academic knowledge of physiological processes but is likely to identify novel therapeutic targets and biomarkers for patient diagnosis, drug safety and efficacy.

1.1.2 Mass Spectrometry

Mass spectrometry (MS) is powerful analytical technique, used to determine the molecular mass of the chemical constituents of a sample. MS is routinely used in a variety of life sciences disciplines to quantify and identify physiologically relevant compounds due to the sensitivity with which the "mass" of a molecule can be measured (Mann *et al.*, 2001). By virtue of the speed of sample analysis, ease of automation and high sensitivity, MS is ideally suited to the high-throughput analysis of complex, chemically-rich biological samples (Glish and Vachet, 2003; Courant *et al.*, 2014).

The mass spectrometer is comprised of three main components – an ionization source, a mass analyser and a detector (*figure 1*; Glish and Vachet, 2003). In order to "detect" the constituents of a sample, a mass spectrometer initially ionizes the constituents, generating charged ions, in the Ionisation Source. The mass analyser separates the gas-phase ions, while the detector is used to measure the mass-to-charge ratio (m/z) and the abundance of the ions formed. In order to prevent the collision of the ions with environmental gaseous molecules, the mass analyser, detector and ion source typically operate under high-vacuum conditions.

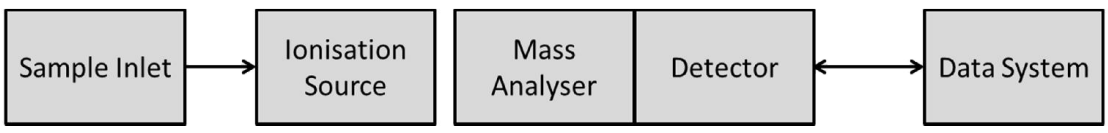


Figure 1: Overview of Mass Spectrometer Components (Adapted from Berdie Rabanaque *et al.*, 2012)

The mass of a molecule is defined by its molar mass, therefore, the term "mass spectrometry" is, in part, a misnomer as the instrument itself measures the mass-to-charge ratio (m/z) of gas-phase ions as opposed to a molar mass (Glish and Vachet, 2003). However, it should be noted that the term "mass" is commonly used to refer to the m/z ratio of an ion when referring to MS data. As the instrument measures the m/z ratio of gas-phase ions the effects of isotopes must also be considered during the analysis. This distinction is highlighted by Glish and Vachet (2003), who describe the example of chlorobenzene (molecular weight 112.56) which may be measured as two distinct ions (m/z 112.01 and 114.01 respectively) with intensities commensurate to the relative abundance of two chlorine isotopes (^{35}Cl and ^{37}Cl).

The ions detected by the instrument's detector during a single scan are typically displayed as a mass spectrum (*figure 2a*). A mass spectrum is a two-dimensional plot of the abundance of an ion (referred to as "intensity") versus its m/z (Glish and Vachet, 2003; Berdie Rabanaque *et al.*, 2012). Due to their rich chemical composition, the constituent molecules of biological samples are typically separated using chromatographic methods prior to MS analysis (*see section 1.1.3*). Therefore, several MS scans may be performed consecutively as the sample elutes. The total ion current (TIC) chromatogram is a plot of the retention time and the summed intensity of the ions, regardless of their m/z ratio, detected at each time point (*figure 2b*).

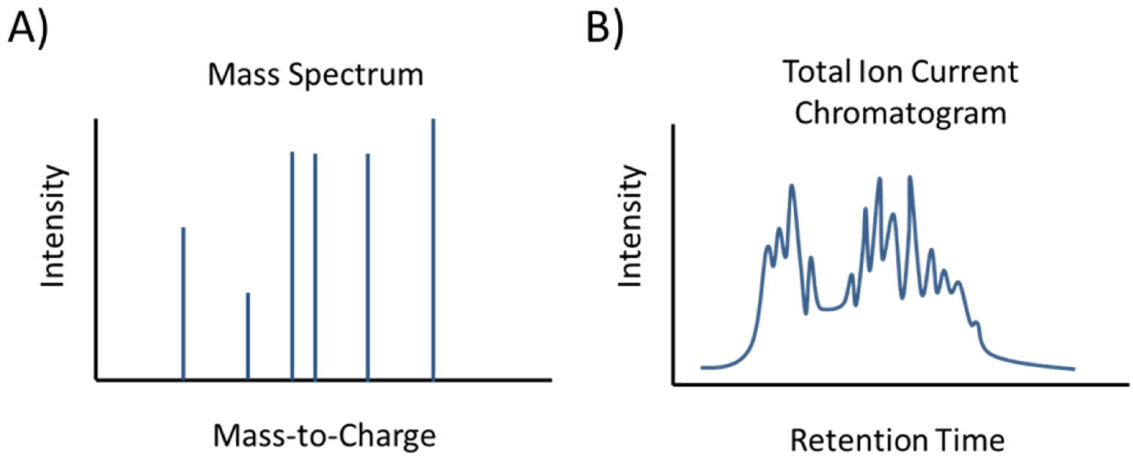


Figure 2: Comparison between mass spectrum (A) and total ion current chromatogram (B).

1.1.3 Chromatographic Methods

As the size of the metabolome is unknown, there may be tens or hundreds of metabolites within a given biological sample with the identical molecular mass. Therefore the ability of the MS instrument to resolve individual metabolites with near identical m/z ratios becomes compromised. The constituents of a biological sample can be separated based upon their chemical or physical properties. By distributing the chemical constituents of the sample between a stationary and mobile phase, the constituents can be separated based upon their chemical or physical properties. The mobile phase, typically a gas or liquid, flows around the stationary phase, either a liquid or solid, and the compounds are separated based upon their relative affinities for the stationary phase. The retention time is a

measure of the time taken for an analyte to elute from a chromatographic column. While the inclusion of chromatographic methods improves the resolution of instrumentation measurements of ion m/z, one limitation is that the retention time may vary between experimental replicates. Gas chromatography-MS (GCMS), referring to a gas mobile phase, is more suitable for small, nonpolar and volatile compounds. Conversely, analysis of polar or ionic metabolites may be achieved with Liquid Chromatography–MS (LCMS; Courant *et al*., 2014). If a gas chromatographic separation method is used for analysis, the addition of a derivatization reaction step prior to sample injection may be considered in order to reduce polarity and increase volatility (Courant *et al*., 2014). The most common derivatization procedures are alkylation, acylation, or silyation, the active hydrogen in functional groups (-COOH, -OH, -NH, and –SH) are replaced by acyl-, or silyl-groups to form esters or ethers (Courant *et al*., 2014).

1.1.4 Ionization

As previously stated, in order for the analytes of interest to be measured by the mass spectrometer they must first be converted to gas-phase ions during at the Ionisation Source. Ionisation can be achieved using various techniques; however, Electrospray Ionisation (ESI) and Electron Impact Ionisation (EII) will be highlighted due to their relevance to the current project.

ESI is regarded as a very 'soft' ionization technique, referring to the limited fragmentation of the sample analytes which occurs using this method, commonly used in conjuncture with liquid chromatography. As such, ESI which allowing non-covalent complexes (such as interacting proteins) to be ionized intact thereby vastly expanding the number of biological applications. One limitation of ESI is the propensity for ion suppression, which can occur in samples with high salt concentrations (>1 mM) or due to the presence of analytes with a high concentration. Ion suppression refers to a diminished ionization efficiency, which reduces the number of ions formed during ionization and subsequently confounding detection.

EII (also referred to as "electron ionization"), is used alongside gas chromatography to ionize and fragment metabolites prior to MS analysis. A typical ionization source exposes the analyte to a stream of thermionic electrons produced from a heated element. Close passage of highly energetic electrons to the neutral analyte induces ionization. These cations (positively charged ions) formed within the ionization source can then be expelled from the ionization source using a repelling voltage. As such, the ions generated from the MS analysis of GCMS EII are exclusively positive in polarity. In contrast to ESI, EII is known as a "harsh" technique because of the degree of fragmentation induced. While the energy of the ionizing electrons can be reduced to diminish the degree of ionization and fragmentation, the sensitivity of the MS instrument to detect the analytes will be negatively influenced.

1.1.5 Mass Analysers and Detectors

Principally, there are five main types of mass analyser in circulation which can be broadly considered in two categories, beam analysers and trapping analysers (Glish and Vachet, 2003). The role of the mass analyser is to separate ions by m/z ratio prior to detection. In the former, the ions from the ion source pass through the analysing field, which serves to separate ions, to the detector in a

beam. Conversely, trapping analysers trap the ions in the analysing field, and are subsequently passed to the detector.

There are three distinct beam analysers. The time-of-flight (TOF) analyser is the simplest, separating ions based on their velocity. Using a fixed potential between the ion source and detector, ions with a lower m/z achieve a greater velocity than those of greater m/z. As ions with the same charge obtain the same kinetic energy, the duration of time taken to travel from the ion source to the detector is used to determine ion mass (Glish and Vachet, 2003). The sector analysers rely upon a similar principle. Analysis of m/z is achieved as ions with the same kinetic energy-to-charge ratio follow an identical path through a magnetic field and are then separated according to their momentum-to-charge ratio in a magnetic sector (Glish and Vachet, 2003). Finally, the quadrupole analysers use radio frequency and direct current voltages applied to four rods (Glish and Vachet, 2003). As ions from the source pass through the mass analyser, those with the same m/z follow a distinct, stable trajectory to the detector from those with a different m/z ratio. Ions of distinct m/z can be sequentially directed to the detector by varying the magnitude of the radio frequency voltages and direct current voltages applied to the four rods (Glish and Vachet, 2003).

While the quadrupole mass analysers maintain electric fields in two dimensions, the quadrupole ion trap (a trapping analyser) maintains electric fields in three (Glish and Vachet, 2003). This quadrupole ion trap analyser therefore can maintain ions within a stable trajectory within the instrument. In contrast to the quadrupole mass analysers, the ions within the quadrupole ion trap do not simply pass through the mass analyser in a mass-selective trajectory. Measurement of the m/z is achieved by making the ion trajectories unstable in a mass-selective manner, which allows for progress to the detector (Glish and Vachet, 2003). In the Fourier-transform ion-cyclotron resonance (FT-ICR) analyser, ions oscillate in a magnetic field at frequencies related to their m/z ratio. As the ions oscillate in close proximity to two metal plates, an alternating current is induced which can be used to derive the m/z ratio (Glish and Vachet, 2003).

1.1.6 Fragmentation

Fragmentation refers to the process in which a parent ion is dissociated, or cleaved, generating product ions which correspond to sub-structures of the parent ion. While fragmentation of ions occurs during the initial ionization of the sample's constituents within the Ionisation Source, this is typically an undesirable consequence of the process necessary to generate measurable ions. Nevertheless as the identity of the compound corresponding to a distinct peak is typically unknown in metabolomic profiling studies, fragmentation of precursor ions can be used to provide valuable structural information which aids in the identification of the compound which formed the parent ion.

The fragmentation spectra of a parent ion can be considered analogous to a chemical fingerprint, or mapping of its chemical structure. As the product ions originate from the dissociation of the parent ion, each product ion corresponds to a sub-structure of the parent ion. However, it should be noted that not all of the parent ion's chemical structure is represented in the fragmentation spectra as the product ions correspond to only those fragments which have maintained a charged state following dissociation. This residual loss of mass is referred to as

the neutral loss. Within the structure of a chemical compound, distinct substructures may have a greater propensity for cleavage than others. Compounds are often classified based on homologous core structures that often will either form a specific product ion or will be lost as a neutral fragment in the MS/MS experiment. As such, under identical dissociation conditions, the pattern of fragmentation of a parent ion is reproducible and can be used to identify functional groups or the compound itself.

In order to garner this valuable resource of structural data, MS can be performed using a technique referred to as Tandem MS (MS/MS). MS/MS may be conducted within either a single instrument (referred to as tandem-in-time) or within interconnected MS instruments (referred to as tandem-in-space). Whether tandem-in-time or tandem-in-space MS is performed is dependent on the capabilities of the instrumentation used in the investigation. Regardless, the principles are the same in each approach.

During an initial full-scan MS stage, selected ions of a specified m/z are isolated from the residual ions originating from the ion source. These isolated ions (the parent ions) are induced to undergo a chemical reaction which induces their cleavage, generating the fragments. The resulting ions from the reaction, termed product ions, are analysed in a subsequent MS stage. Typically, MS/MS simply corresponds to two stages of MS analysis, the full-scan of the sample ions and the analysis of the fragmentation product ions. However, in tandem-in-time MS/MS instrumentation it is possible to repeat the fragmentation process n times. As such, the product ions of a precursor ion could themselves be fragmented, generating a further generation of product ions for analysis.

1.1.7 Dissociation Methods

A crucial aspect of fragmentation experiments is the reaction that occurs to induce dissociation of the parent ion. The most frequently used reaction is unimolecular dissociation, which is generally enhanced by some form of ion activation. Ion activation is necessary to increase the internal energy of the parent ion so it will dissociate prior to analysis by MS2. In practice, activation and dissociation cannot be separated, so the ion activation methods are simply referred to as dissociation methods. The method almost universally used is collision-induced dissociation (CID), which occurs within a distinct collision zone of the MS instrument. In CID, the parent ion collides with a neutral target (collision) gas and some of the kinetic energy of the parent ion can be converted to internal energy.

1.1.8 Data Dependent Acquisition (LCMS)

Liquid chromatography MS/MS can be performed under information or data-dependent acquisition conditions. This type of acquisition is termed "auto-adaptive MS/MS product-ion scan mode", in which parent ion selection during the full-scan mode (termed the "survey scan") is dependent upon predetermined criteria (Marquet *et al.*, 2003). Selection of the parent ions for dissociation is typically dependent on the intensity of the ions in the survey scan. The most intense ions are transmitted to the collision cell, where fragmentation occurs. The resulting product ions are then analysed by a second MS phase (Marquet *et al.*, 2003; *figure 3*).

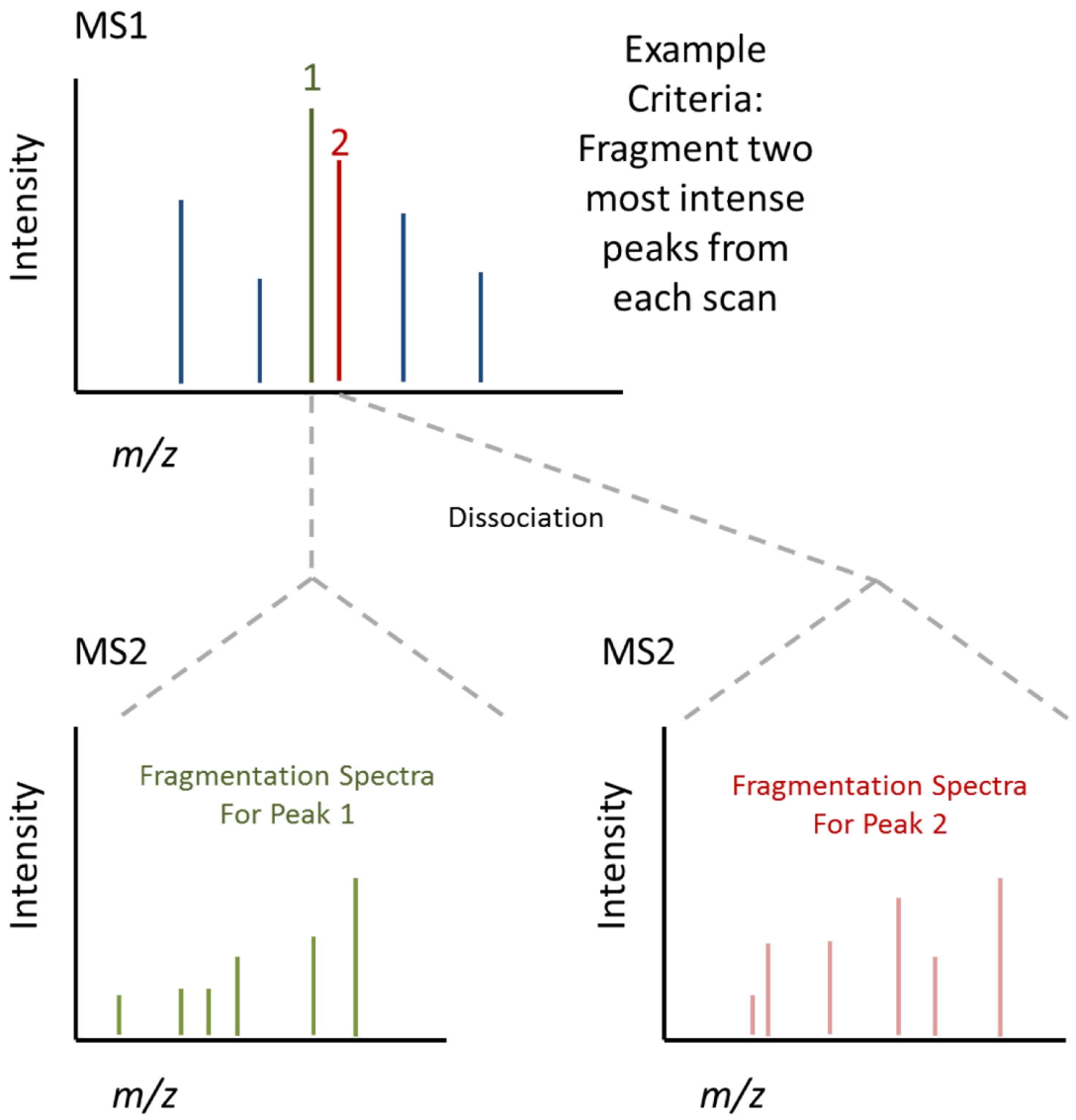


Figure 3: Data-dependent acquisition.

1.1.9 Data Independent Acquisition (LCMS)

Whilst highly-powerful, one limitation of the data-dependent acquisition technique is that selection of parent ions during the initial survey scan favours metabolites with higher ionization efficiencies (Chapman *et al.*, 2014). In order to circumvent this bias and increase the detectable dynamic range, the concept of data-independent acquisition was proposed (Chapman *et al.*, 2014). In data-independent acquisition, there are no intensity-based criteria for ion selection based on prior scans. Instead, a predefined m/z range is investigated by fragmenting all ions entering the mass spectrometer at a given retention time (termed "broadband DIA"; Chapman *et al.*, 2014). Alternatively, the m/z range can be divided into smaller, discreet m/z ranges for isolation and subsequent fragmentation. Therefore, the fragmentation of precursor ions is independent of any prior data generated from the sample. Due to the fragmentation of all ions entering the mass spectrometer, precursor-product ion lineage is lost in DIA.

However, distinct experimental approaches can be implemented to circumvent this apparent limitation. In the MS^E technique, scans using a Q-TOF instrument can cycle between 'high-energy' and 'low-energy' scans. While the 'high-energy' scan generates the fragmentation data, the 'low-energy' scan can be considered analogous to the 'survey scan' of data-dependent acquisition (Egertson *et al.*, 2015; *figure 4*). Therefore, the lineage of parent and product ions can be resolved.

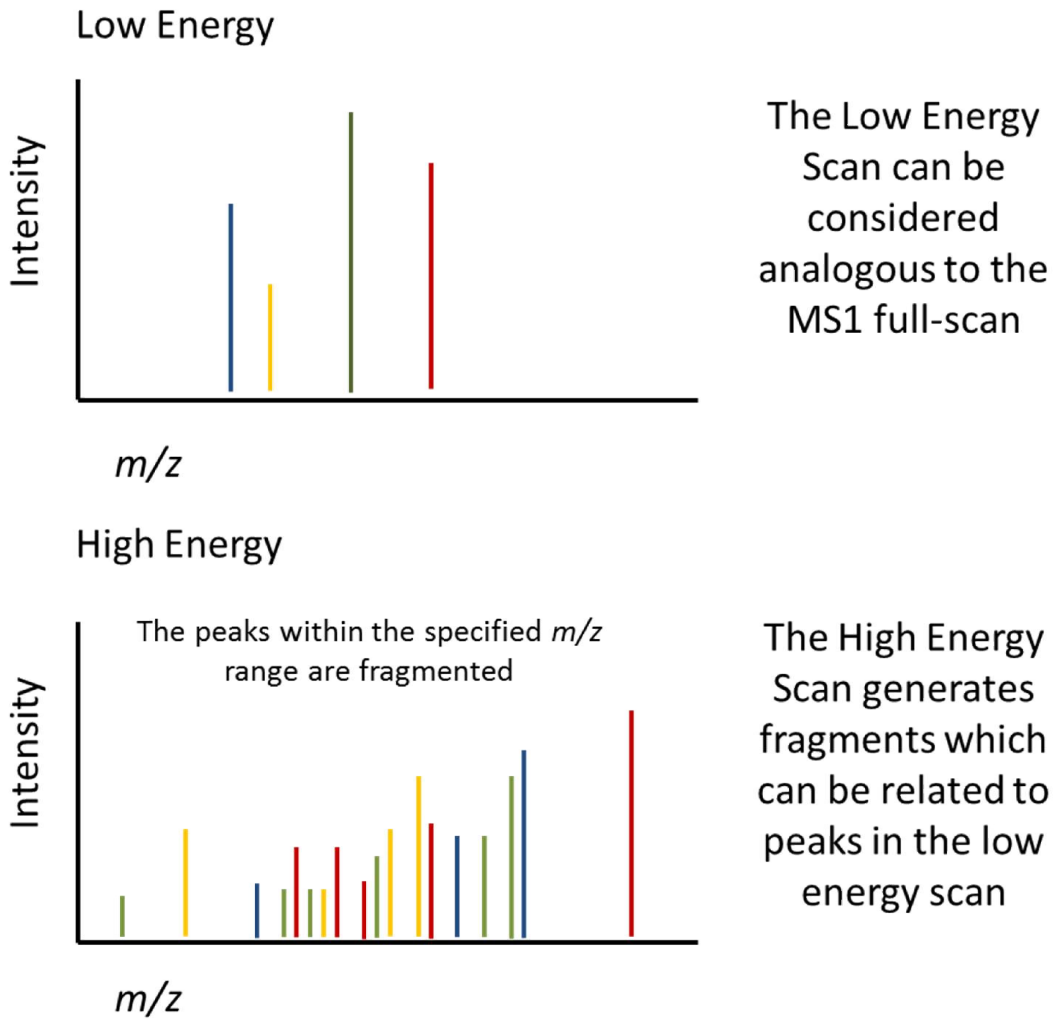


Figure 4: Data-independent acquisition.

1.1.10 Gas Chromatography-MS Electron Ionisation

As described in section 1.1.4, EII is a 'harsh' ionisation technique typically used prior to the GCMS analysis of samples. Contrasting the use of ESI in LCMS experiments, analytes exposed to EII typically fragment prior to MS analysis and therefore fragmentation does not occur in the collision zone of the MS instrument as it does for both the data-dependent and data-independent acquisition methods detailed previously. As such, the full-scan of a GCMS analysis consists of product ions generated from the analyte (the parent molecule). In short, the MS instrument does not "detect" a parent ion for the fragmentation spectra generated using GCMS. Nevertheless, the product ions detected by the

instrument originate from an 'anonymous' product analyte during the fragmentation of the sample. The fragments themselves may still be grouped, which in turn allows for identification of the parent analyte (*figure 5*).

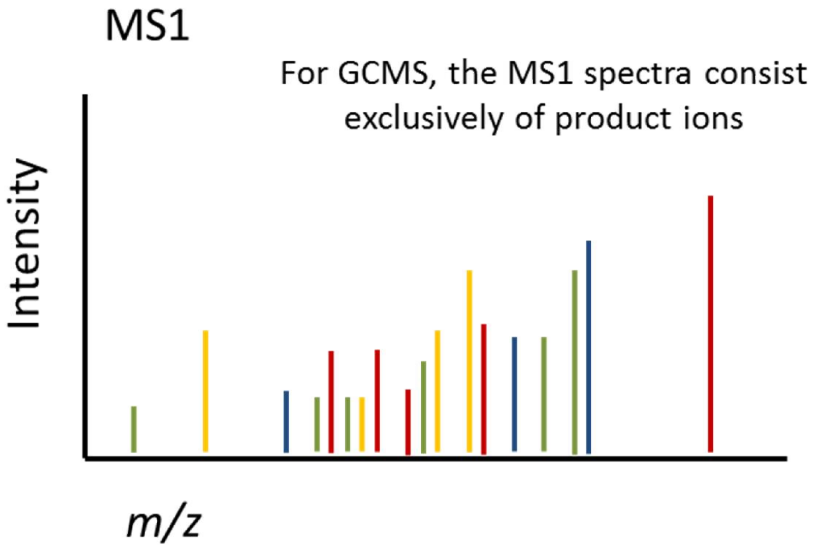


Figure 5: Gas-Chromatography Electron Impact Ionisation

1.1.11 Data Analysis

Typically, an MS-metabolomics experiment generates vast quantities of data. The processing of such datasets manually would be incredibly time-consuming and error-prone. Therefore, specific software tools and algorithms are essential to the fulfilling the aims of metabolomics – quantification and identification. Initially, the raw data, output by the MS instrumentation must be converted from the instrumentation providers proprietary file format into a format suitable for either statistical analysis or compound identification.

The ProteoWizard Library (http://proteowizard.sourceforge.net/) is an open-source collection of software tools, initiated in 2007, to allow for easy conversion of vendor-specific MS file formats to open-source MS-specific XML formats. The aim of the project was to ensure that researchers focus upon the development of novel analytical approaches as opposed to developing file conversion software. While several open-source file formats are available (such as mzData, JCAMP-DX and ANDI-MS), one commonly adopted format is the mzXML which is beginning to be superseded by the mzML format. Nevertheless the mzXML format, an XML based format, has maintained popularity in both proteomics and metabolomics MS studies. As two formats (mzXML and mzData) were undesirable, attempts have been made to generate a unified format, referred to as "mzML", which aimed to incorporate the most desirable features of both formats.

At present, many proteomics and metabolomics spectral reference libraries are accessible via a web client and remain the primary source used in compound identification from fragmentation spectra (Go, 2010). Theoretical hits, or

candidate annotations, are returned to the user in a tabular fashion upon the submission of fragmentation spectra in an appropriate format. The fragmentation spectra generated from an experiment are compared to those stored in the library of reference compounds and are returned with a measure of confidence. The unknown metabolite is typically identified by the experimenter from candidate reference spectra which best match that of the unknown metabolite. Representative spectral libraries commonly used in metabolomics include the resource library from the US National Institute of Science and Technology (NIST), the Golm Metabolite Database (GMD), MassBank, METLIN and the Madison Metabolomics Consortium Database (MMCD; Go, 2010).

Despite their widespread use in the field of metabolomics, many spectral reference libraries lack suitable documentation, tutorials or help pages for users (Go, 2010). In addition, the reference compounds contained in each spectral reference library vary, and therefore the querying of numerous resources may be required to identify appropriate candidate annotations for a given peak. However, both the compound name and formula are unreliable for the purposes of comparing hits between libraries. As such, attempts to standardize compound identification has led to an abundance of distinct codes, such as the IUPAC International Chemical Identifier (InChi) and the Chemical Abstracts Service (CAS) Registry Number, which may serve as unique compound identifiers and allow for comparison between reference libraries.

Chapter 2 Requirements

2.1 Problem Definition and Scope

Metabolomics aims to quantify and identify metabolites of physiological significance within biological systems. At present, the latter poses significant challenges to researchers within the field, due to a lack of suitable software tools, algorithms and the volume of data generated. The aim of the project was to develop a web-based application capable of generating candidate annotations using fragmentation spectra derived via distinct experimental protocols in order to aid researchers in metabolite identification. As such, the application must maintain the lineage of parent and product peaks, in order to perform queries of spectral reference libraries. While the application should standalone, it is intended to be deployed within an existing proprietary pipeline in order to provide supporting evidence for future scientific communications and improve the efficiency of existing services.

2.2 Client and Users

2.2.1 The Client

The application is to be developed for the Metabolomics Facility of Glasgow Polyomics. Glasgow Polyomics is a research centre within the College of Medical, Veterinary and Life Sciences of the University of Glasgow situated within the Wolfson Wohl Cancer Research Centre at the Garscube Campus. The Metabolomics Facility provides services including the untargeted analysis of polar metabolites and verification of compound identities using reference standards. In addition, the facility supports academic research staff with broad research interests within the field of metabolomics. The primary point of contact was Dr Karl Burgess, the Head of Metabolomics (hereafter referred to as the client).

2.2.2 The Users

In addition to the primary point of contact, several members of staff were available during requirements gathering and for the evaluation of prototypes. As such, they imparted invaluable knowledge of both the domain and the needs of the users. Of note were Yoann Gloaguen (Deputy Metabolomics Laboratory Manager and primary developer of the existing pipeline), Dr Ronan Daly (Data Analyst Manager), Dr Stefan Weidt (Mass Spectrometry Technologist) and Dr Justin Van Der Hooft (Mass Spectrometry Technologist).

The intended users of the application are the research staff within the Metabolomics Facility. The users will have an advanced and detailed knowledge of the scientific domain and familiarity with domain terminology. However, technical proficiency is likely to vary. Some members of the research staff having a strong computational background within the field of Bioinformatics, whilst those from a bench-top research background may have less familiarity. Nevertheless, all users will be familiar with use in web applications due to familiarity with the existing pipeline.

2.3 Requirements Gathering

An overview of the problem domain and an introduction to the relevant technologies were provided in a series of meetings with both Dr Simon Rogers and Joe Wandy at the onset of the project. This provided the impetus for the initial research to garner sufficient knowledge of the necessary terminology prior to the client brief.

The system requirements were elaborated upon in significant detail during an initial meeting with both the client and the primary developer of the existing pipeline, Dr Karl Burgess and Yoann Gloaguen, respectively. Based upon the client brief, the functional and non-functional requirements of the proposed system were documented (*Appendix B*). In turn, the primary user (a Scientific Researcher) was identified and a set of use cases were defined and prioritised using the MoSCoW methodology to refine the scope of the project. However, the priority and content of the use cases were reviewed incrementally following the demonstration of prototypes to the client and in agreement with the project supervisor.

2.4 Environment

2.4.1 Summary of PiMP Functionality

The existing analytical pipeline consists of a Django-based web application named PiMP, which allows users to create, modify and share MS projects from their web browser. In addition to sample files, the user can upload calibration files such as blanks, quality control (QC) and standard files in either the .mzXML or .csv file formats. Upon completion of the file upload, the user may specify pairwise comparisons between experimental conditions for an analysis. Using R, the pipeline calls scripts to extract peak data from the source files which are displayed to the user in the form of an interactive TIC chromatogram. From which, the user can view the mass spectrum at a given retention time by clicking on the corresponding data point in the TIC chromatogram. Candidate annotations are derived for the peaks identified in the mass spectrum. However, the veracity of candidate annotations in the existing pipeline is limited. Peak data cannot be stored in a hierarchical manner within the existing database schema, thereby preventing the utilization of fragmentation data. Therefore, PiMP supports the analysis of peak data obtained from the 'full-scan' of the initial MS phase, but not MS/MS. As such, the inability of the application to maintain the lineage of the parent and precursor peaks has limited its extensibility to incorporate data generated by additional experimental protocols.

2.4.2 Hardware and Software Requirements of PiMP

The existing application, based in Django 1.7, runs on a standard (non-specialized) server which runs a Linux-based operating system. The data for the application is stored in a concurrency-supporting MySQL database. Within the application, concurrency within the application is supported through the use of django-celery (Celery Project, 2015), which performs the computationally intensive tasks added to a queue via a RabbitMQ message broker (Pivotal, 2015).

The extracting of peak data from the source files is performed, using R scripts, as a background process to maintain the responsiveness of the web server. The R scripts are dependent upon the following standard R packages: 'RUnit', 'DBI', 'RCurl', 'RJSONIO', 'XLConnect', 'outliers', 'gptk' and 'doParallel'. In addition, the R scripts require the biocLite R packages 'impute' and 'limma' and the installation of the 'mzmatch.R' package. The 'mzmatch.R' package integrates both mzMatch and XCMS, which are open-source software packages for the analysis of metabolomics data sets. Instructions for the installation of 'mzmatch.R' and its dependencies can be found online (mzMatch, 2015).

The research staff have access to desktop computers, which predominantly run on Windows operating systems and have access to an array of internet browsers including Firefox, Google Chrome and Internet Explorer. However, the existing PiMP application has been optimized for Internet Explorer and therefore its use by the staff has been encouraged by the current developers.

2.5 Competing Systems

Existing software applications currently available for the analysis of MS datasets include both proprietary and open-source systems. These have been developed by commercial enterprises or academic institutions and are typically available as either GUI-based desktop clients or web services. To allow for comparison with the proposed application, examples of web-based systems for the generation of candidate annotations will be discussed to emphasise their limitations and the requirement for development of proposed application.

ALLocator is a freely-available web application for the quantification and identification of metabolites (Kessler *et al.*, 2014). The ALLocator web platform supports the upload of MS raw data in the .mzXML, .mzML, and .CDF formats (Kessler *et al.*, 2014). Using the centWave LC-MS feature detection method of the XCMS R package, a peak list is derived (Kessler *et al.*, 2014). The application provides two distinct tools for spectra deconvolution. Spectra deconvolution refers to a process which aims to improve the resolution of the measured peaks, through the removal of artifacts introduced by the instrumentation (Marchetti and Mignerey, 1993). Using the fragmentation spectra, the application returns candidate annotations from the Kyoto Encyclopedia of Genes and Genomes (KEGG) repository. The application has been developed specifically to support the analysis of LCMS-ESI datasets, and therefore does not support the additional methodologies requested by the client. Furthermore, spectral queries are limited to a single repository, the Kyoto Encyclopedia of Genes and Genomes (KEGG) reference library, thereby diminishing the total number of available reference spectra.

A similar limitation has been identified for the Competitive Fragmentation Modeling for Metabolite Identification (CFM-ID) web server (Allen *et al.*, 2014). Services include the annotation of peaks for a known chemical structure and the ranking of candidate structures for a spectrum (Allen *et al.*, 2014). For the identification of the compounds, the server allows for the querying against uploaded user-specified structures (up to a maximum of 100), the Human Metabolome Database (HMD) and the KEGG libraries (Allen *et al.*, 2014). While more comprehensive coverage of potential metabolites is achieved by the

inclusion of HMD, the volume of reference spectra may still be insufficient to fulfill the needs of the client.

In contrast to CFM-ID and ALLocator, the MetaboAnalyst web-service provides support for both GC and LC-MS datasets (Xia and Wishart, 2011). A unique feature is the inclusion of multivariate statistical analysis, which is desirable but is not, as yet, available within PiMP (Xia and Wishart, 2011). In addition, the service provides extensive tutorials to guide inexperienced users through the pipeline. Although peak annotation and putative pathway identification are provided, the predominant aim of the service is to support quantification. As such the developers of MetaboAnalyst acknowledge the service has limited annotation functionality, due to an inability to process raw spectral MS data files (Xia and Wishart, 2011). Furthermore, the service requires partially analysed MS data as input, such as a peak lists, therefore inexperienced users are required to perform processing of the data in advance.

2.6 Existing Procedures

At present, the retrieval of candidate annotations is dependent upon the experimental protocol. For LCMS, PiMP can be used to retrieve candidate annotations based on the "full scan" data of MS1 alone. However, it is the veracity of annotations retrieved by PiMP the client aims to improve with the utilization of fragmentation spectra. Alternatively, experimenters may format the fragmentation spectra of an analysis appropriately, and directly query an online or locally installed spectral reference library. However, this process is labor intensive, error-prone and requires knowledge of the user interface of each distinct library to be queried which may be problematic for inexperienced members of staff.

2.7 Summary of Functional Requirements

The application must allow authorized users, to define a fragmentation experiment and experimental samples. In turn, the application must allow for user to upload MS data files to an experimental sample. The experimental samples should be grouped by the application within distinct experimental conditions defined by the user. From the uploaded sample files, the application must allow the user to extract peak data (including the m/z, retention time and intensity), which must be stored in the database in a hierarchical manner. The application must display fragmentation spectra to the user in a graphical format. The application must allow authorized users to retrieve or generate candidate annotations for the peaks extracted from the source file. The application must store the candidate annotations, data for the corresponding chemical compounds and a measure of 'hit' confidence. Furthermore, analysis performed in the PiMP application must generate, retrieve and display candidate annotations from the application for those fragmentation data files uploaded to the PiMP application.

2.8 Summary of Non-Functional Requirements

The application must be developed using the Django framework (version 1.7) and be compatible with the existing server and database management system (MySQL) implemented by the client. The application must support the mzXML

file format, but could also support the mzML file format anticipating future needs. Due to the aspirations of the client to develop the application further, the application must be capable of future enhancement. As a Greenfield project, prototypes of the application should be demonstrated to the client at regular intervals. The application is to be delivered to the client via a local Git repository, no later than the 07/08/2015.

2.9 Use Case Summary

During the requirements gathering process, a single user was identified. A scientific researcher will have extensive domain knowledge. The following use cases were attributed to the scientific researcher whose motivation is to identify the metabolites detected within an MS analysis of a biological sample. For additional detail, descriptions of the 'key' use cases are provided in Appendix B.

A Scientific Researcher **MUST** have the ability to…

- Create and modify a fragmentation experiment
- Create and modify the experimental samples within an experiment
- Upload MS data files to an experimental sample
- Derive peak data from uploaded sample files
- Generate or retrieve candidate annotations for MS peaks
- View fragmentation spectra
- Generate fragmentation spectra from data files uploaded to the PiMP application.
- Generate and view candidate annotations for peaks derived from data files uploaded to the PiMP application.

A Scientific Researcher **SHOULD** have the ability to…

- Group experimental samples by experimental conditions.
- Specify a preferred candidate annotation for a peak.
- Share a fragmentation experiment with other authenticated users.

A Scientific Researcher **COULD** have the ability to…

- View the chemical structure of the compound associated with a candidate annotation.
- Visually compare the measured fragmentation spectra, with that stored in a spectral reference library.
- Consolidate peak data derived from several experimental replicate data files.
- Extract and consolidate peak data from distinct MS1 and MS/MS data files.
- Specify a method for the extraction of peak data from source files.

A Scientific Researcher **WOULD** have the ability to…

- Export peak data and candidate annotations.

Chapter 3 Design

3.1 Design Approach

Upon completion of the requirements gathering and documentation phase of the project, an evaluation of risks and uncertainties was performed to identify potential pitfalls (*section 3.2*). Throughout the design process, the Fragment Annotation Kit (FrAnK) application was designed based upon the identified use cases (*section 2.9*). However, the iterative approach adopted during development necessitated the evolution of designs in response to feedback garnered by the demonstration of prototypes to both the client and users. To ensure integration with PiMP and compatibility with the existing infrastructure, the N-tier architecture of the current pipeline was maintained and therefore was not re-evaluated.

The design process initiated with the creation of an entity-relationship diagram (*section 3.3*). The aim was to model the data to be stored and provide a foundation for the creation of an initial database schema. In addition, this provided clarity which helped overcome an initial lack of domain knowledge at the onset of the project. Using the entity-relationship diagram as a guide, an initial database schema, detailing the format of the requisite fields, was generated (*section 3.4*). The design phase then transitioned to focus upon the site- and URL-mapping, detailing the user navigation through the application (*section 3.5*). From which, wireframes were designed to consider how the information stored could be presented to the user in a clear and logical manner (*section 3.6*).

3.2 Risks and Uncertainties

Upon a review of the PiMP application, it was identified that numerous distinct technologies had been implemented within the existing pipeline which I, as the developer, had little to no familiarity with. While I had modest experience using the Django framework, I was unfamiliar with Celery and the programming language R. Furthermore, the integration of these technologies posed a significant risk to the development of FrAnK. In addition, the querying of the spectral reference libraries represented a risk due to the variety of the technologies (varying in both the format and data content of both inputs and outputs) implemented for batch processing of fragmentation spectra and the clarity of the associated documentation.

The project also presented a significant requirements risk, due to a lack of advanced domain knowledge. However, the use of use case and entity-relationship modelling, study of relevant literature, demonstration of prototypes and regular communication with the client, users and project supervisor were used to diminish the impact of this risk upon development.

3.3 Entity-Relationship Modelling

To gain familiarity with the domain and provide clarity to the design of the database schema, an entity-relationship diagram for FrAnK was prepared (*figure*

6). A review of the PiMP database schema provided a starting point and was invaluable in the identification of entities and their relationships. In addition, the process of review provided insight into the analytical processes of the existing pipeline, its limitations and points of potential integration for FrAnK.

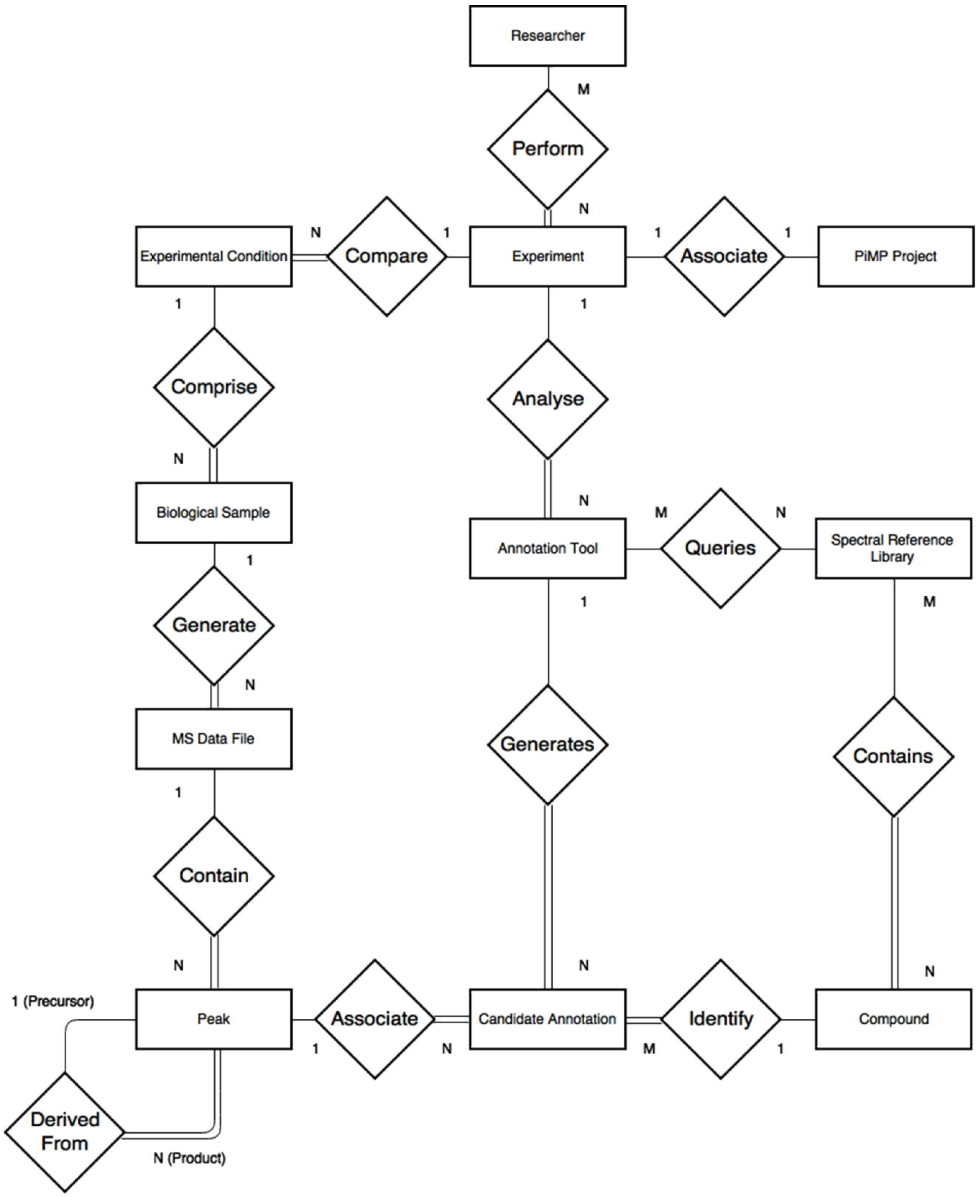


Figure 6: Condensed Entity-Relationship Model

The researcher performs experiments which consist of the comparison between experimental conditions, which in turn may be comprised of several biological samples. The analysis of a biological sample using MS may generate numerous data files, depending upon the experimental protocol adopted. Each data file contains numerous peaks measured by the MS instrumentation. The experimenter performs an analysis of the experiment using an 'annotation tool', which generates candidate annotations for each peak, by querying of the fragmentation spectra in a spectral reference library. Each candidate annotation, labels each peak with a putative chemical compound identification. A chemical compound may originate from one or many distinct spectral reference libraries. Finally, each experiment in FrAnK may be associated with an experiment in

PiMP. This would allow for integration between both the PiMP and FrAnK applications.

While the entity-relationship diagram provided the foundation for an initial database schema (*Appendix C*), several limitations were identified which were addressed during the development process.

3.4 Database Design

As alluded to in the previous section, the mapping of the entity-relationship model to a database schema (*Appendix C*) unveiled several initial assumptions which would otherwise diminish extensibility.

While an experiment is generated by a single researcher, an experiment may be of interest to several of the research staff. As such, an experiment may have collaborators who wish to perform administrative tasks on the experiment (such as upload additional data files or include additional experimental conditions) or perform additional analysis of the fragmentation spectra. Therefore, a many-to-many relationship was added to the experiment table to facilitate the addition of collaborators to an experiment in subsequent iterations of FrAnK (*figure 7*).

The entity-relationship model identified that peak data is derived from the MS data files. While this is indeed the case, the entity-relationship model failed to reflect that there is not a single, but numerous distinct methods for the extraction of peak data from a source file. This may be due to the varying pick-picking criteria, filtering or the use of distinct software packages to derive the peak data. As such, the content of the peak data generated from the same source files may vary depending upon the algorithm implemented. To resolve this issue, the database design introduced the concept of a 'Fragmentation Set', which can be considered as the grouping of the peaks extracted from the source files of a single experiment (*figure 7*). This added greater extensibility to the application, as a single source of MS data may generate numerous distinct fragmentation sets for analysis.

An additional modification was the inclusion of an 'Annotation Query', which is analogous to the 'Annotation Tool' identified in the entity-relationship model. The 'Annotation Query' is a user-generated request for the annotation of the peaks contained within a single 'Fragmentation Set' (*figure 7*). As such, a query is now associated with a single 'Fragmentation Set' as opposed to the 'Experiment' itself (*figure 7*). An 'Annotation Query' generates candidate annotations through the use of an 'Annotation Tool' (formerly referred to as the "Spectral Reference Library" in the original model), passing parameters specific to the query. The inclusion of a many-to-many table ('Annotation Query Hierarchy') will allow for the development of annotation tools which take the candidate annotations retrieved from one or many parent 'Annotation Query' instances to derive new candidate annotations, thereby sub-querying an existing 'Annotation Query' (*figure 7*). When the entity-relationship model was initially conceived, it was not considered that such 'Annotation Tools' would be developed.

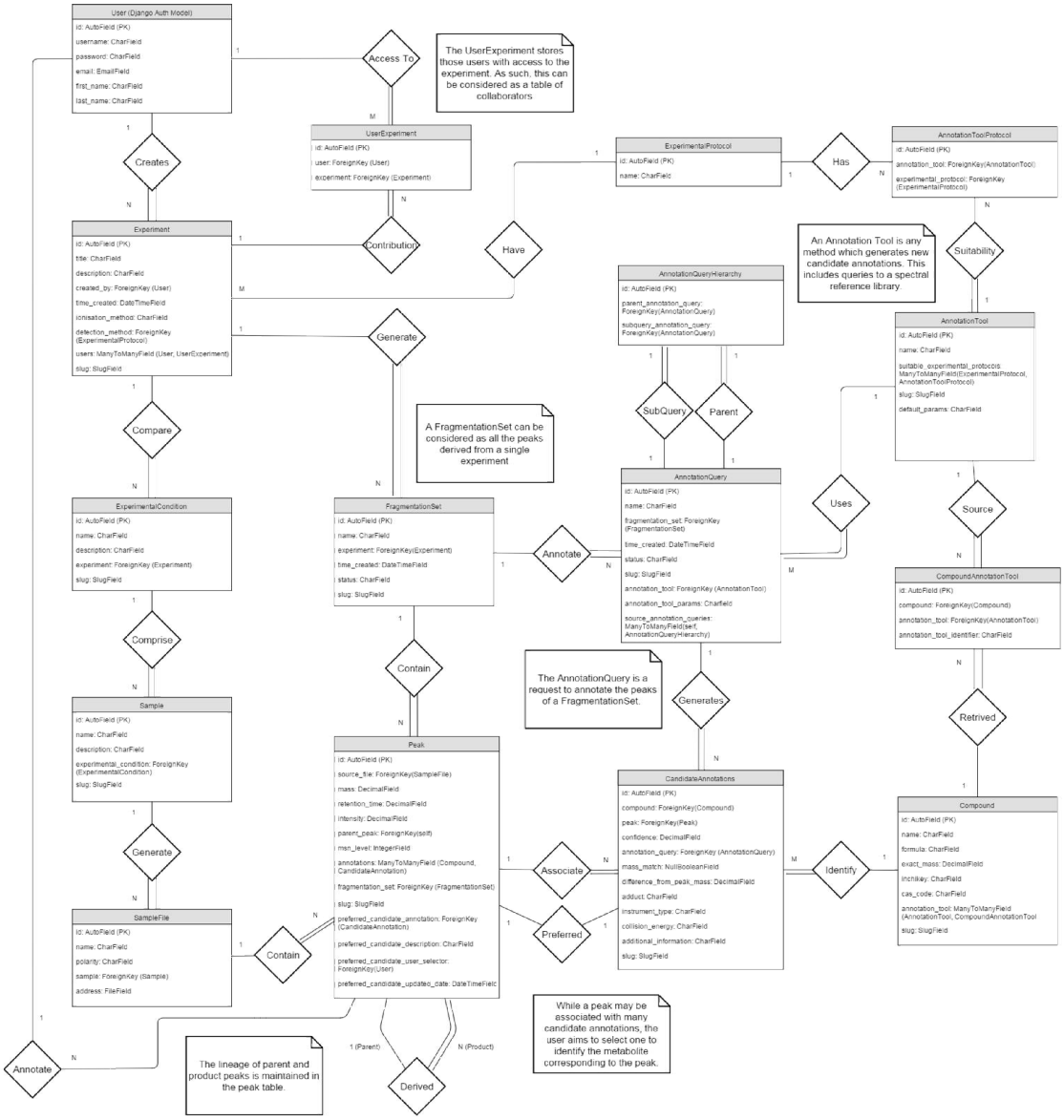


Figure 7: FrAnK Database Schema

This oversite necessitated the renaming of the 'Spectral Reference Library' of entity-relationship model to the generic 'Annotation Tool' (*figure 7*). An 'Annotation Tool' is considered as any software, algorithm or web service which may be queried for the retrieval or generation of novel candidate annotations. Previously, it was assumed that the source of novel candidate annotations in the FrAnK application would exclusively be through the querying of fragmentation spectra to a spectral reference library. However, this unnecessarily constrains the application, and limits the development of novel approaches to metabolite identification. While the querying of spectral reference libraries is appropriate to all experimental protocols, future 'Annotation Tools' may be developed for a specific MS methodology.

In order to facilitate the limiting of an 'Annotation Tool' to a selective MS methodology, an 'Experimental Protocol' table was introduced into the database with a many-to-many relationship to the 'Annotation Tool' table (the 'Annotation Tool Protocol'; *figure 7*). As such, the experimental protocols and the tools

appropriate to the methodology could be designated in a population script for the application.

Finally, the entity-relationship model failed to acknowledge the selection of a preferred candidate annotation for a given peak by an authorized user. As such, the Peak table was modified to include a foreign key reference to a single candidate annotation (*figure 7*). In addition, the user, the justification and time for the selection of the annotation would be stored. During development, it was considered that the inclusion of these additional fields may form a transitive dependency within the database schema. Alternative approaches were considered, such as the inclusion of a 'Preferred' boolean in the candidate annotation table as each candidate annotation is only associated with a single Peak. However, the relationship between the Peak and its preferred annotation is a 1 to 1 relationship, and as such it was decided that the preferred candidate annotation was an attribute of the Peak analogous to the confirmation of its identification.

3.5 Site Map and URL Mapping

Based on the database schema, an ideal site-map was designed for the FrAnK application (*Appendix C*). However, the design was altered during development due to the time-constraints of development. The index page is accessed via the navigation bar of the PiMP application. From which the user, may navigate to pages listing their existing experiments ('My Experiments') and fragmentation sets ('My Fragmentation Sets'). In subsequent iterations of FrAnK the user may access a 'Notifications' page, detailing changes made to their existing experiments by collaborators and the completion of background processes. Furthermore, the application may include a 'Compound Library', which would allow for the user to browse the compound data stored by the application's database.

From the 'My Experiments' page, the user may select or create an experiment. The 'Experiment' page displays the details of the experiment, including any associated experimental conditions and fragmentation sets. In the original design of the site-map, the creation of experimental conditions, samples, fragmentation sets and the drag-and-drop upload of sample files would be accessible from within this page. However, implementation was not achieved in the time available. At present, experimental conditions can be selected or created from the 'Experiment' page. Selection of an experimental condition navigates to an 'Experimental Condition' page which allows for the creation of samples and the upload of data files. Fragmentation sets can be selected or created from the 'Experiment' page, once a data file has been uploaded.

The 'Fragmentation Set' page, also accessible via the 'My Fragmentation Sets' page, displays the MS1 peaks derived from the data files of the associated experiment. The user may create a new 'Annotation Query', in order to generate new candidate annotations for the fragmentation set, upon selection of an 'Annotation Tool' from a drop-down menu provided at the top of the page. Each MS1 peak displayed on the 'Fragmentation Set' page serves as a link to a 'Peak' page.

The ‘Peak’ page provides both a graphical and tabular display of the fragmentation spectra associated with the peak. The table of product peaks provides links to each product peak’s ‘Peak’ page, thereby allowing for navigation through *n* levels of MS/MS data. Furthermore, the ‘Peak’ page provides details for the candidate annotations associated with the peak. At present, the candidate annotation table provides a link to select a candidate annotation as the ‘preferred’ annotation for the peak. However, it is intended that the candidate annotations will serve as links, providing access to a compound page providing additional information.

The URL Mapping for FrAnK is provided in Appendix C. The URL was designed to be as intuitive as possible to improve usability. In addition, the slug for a database entry was derived from the user-specified name where possible (e.g. the title of an experiment or name of a fragmentation set). However, the peaks and candidate annotations do not have intrinsic names. As such, unique identifiers for these database entries were provided as an alternative.

3.6 Wireframes

During the initial requirements gathering phase, the client stated that the back-end functionality, as opposed to the aesthetics, of the application should be prioritised at this juncture. Nevertheless, wireframes were designed for key pages of the application, namely the ‘Experiment’ page, the ‘Fragmentation Set’ page and the ‘Peak’ page (*Appendix C*). While the wireframes are representative of the data stored by the FrAnK application, maintaining a uniform style of presentation to that of the PiMP application was considered a priority. The PiMP application has been successfully implemented by the client, and as such the users are already familiar with the existing layout and navigation of the application. Therefore it was considered that maintaining a similar theme to the PiMP application may increase user adoption and acceptance, reducing the duration of time required by the users familiarise themselves with the novel functionality.

Chapter 4 Implementation

4.1 Development Process Overview

As a Greenfield project and due to the availability of the client for the evaluation of prototypes an iterative approach to development was adopted. Intended to be developed further upon delivery, it was considered that the requirements for the application may require reassessment or reprioritising dependent upon the client feedback to prototypes. Therefore, the development of the application can be considered as two distinct iterations.

FrAnK was developed on an HP 15 Laptop, consisting of an Intel Core i3-45005U processor and 7.7 GB RAM, running the Ubuntu 15.04 operating system. PiMP, including the dependencies described in section 2.4.2, was installed. To provide clarity for future developers, additional packages were installed such as mysql-server 5.6.25, r-base-core 3.1.2-2, wine 1.6.1:1.6.2, rabbitmq-server 3.2.4-1 and oracle-java8 in addition to those stated in the 'requirements_frank.txt' document of the application.

The initial iteration consisted of the development of a rudimentary Django application to simply enable a user to set-up of an experiment and upload MS source files. From this juncture, the implementation transitioned to the extraction of peak data from the uploaded mzXML files. Initially, this consisted of data generated from a single experimental protocol. LC-MS/MS data-dependent acquisition (LC-MS/MS-DDA) was selected due to the frequency with which the protocol is utilised by the researchers of the Metabolomics facility. In order to achieve implementation, the process of peak extraction not only required the interfacing with R, but is performed as a background process using Celery. Upon implementation of LC-MS/MS-DDA peak extraction, the project transitioned to the retrieval of candidate annotations from a spectral reference library. Following an evaluation of the available documentation, the MassBank Web API was selected from those of interest to the client. In part, this was due to the clarity of the API documentation. Nevertheless, as a public library MassBank provides an extensive compound library which would be of value to the users. Implementation of the MassBank batch service was achieved through the packaging of query spectra within a SOAP request. The culmination of the initial phase of development was a prototype application consisting of the primary functionality, namely batch searching of fragmentation spectra, expected of the application. At which point, the prototype of the application was demonstrated to the client and a selection of potential users.

Following the evaluation of the initial prototype, the client requested the implementation of an additional experimental protocol and spectral reference library. As such, the second phase of implementation consisted of incorporating the extraction of peak data generated by the GCMS Electron Impact Ionisation (GCMS-EII) protocol. As before, this was implemented via an interface with R. However, to generate a peak list an R script was created to perform peak extraction and grouping. In addition, the spectral reference libraries of NIST14 were provided by the client for implementation. However, the querying of the libraries is performed by the MS PepSearch software developed for Windows. As

the client's server runs a Linux-like operating system, Wine was used to run the MS PepSearch software. As before, the completion of the second phase culminated in the demonstration of a prototype to the client. Upon which, it was requested that FrAnK be integrated into the existing pipeline in a manner which would allow for retrieval of candidate annotations from within PiMP. However, integration could not be achieved within the allotted development time.

4.2 Application Overview

For those unfamiliar, Django is a web application framework based in Python which implements a Model-View-Template (MVT) design pattern. The PiMP project folder comprises various applications. Each application typically consists of a models.py, views.py, admin.py, urls.py, forms.py and tests.py files and the framework provides compatibility with various database management systems such as MySQL, PostgreSQL and SQLite. For those unfamiliar with the Django framework, the online documentation (Django Software Foundation, 2015) and walkthrough tutorials (Azzopardi and Maxwell, 2013) provide an excellent source of introductory material.

The FrAnK application, encapsulated with a distinct folder of the PiMP project, contains each of the aforementioned python files. The models.py file contains python classes which typically map to a single database table, therefore providing a python representation of the database schema (referred to as the Object-Relational Model). These models are registered in the admin.py file of the application, to allow for the site administrators to access the data stored in the database in a direct manner via the admin interface provided by the Django framework. In addition, the urls.py file contains a tuple of regular expressions which serve to map incoming URLs to an appropriate view. A view, declared in the views.py file, provides the logic associated with each page of the application. Each view serves to return the appropriate html template, including necessary forms and object-relational model instances in response to either a GET or POST request.

In addition to the standard files of a Django application, the FrAnK application includes the tasks.py, annotationTools.py and peakFactories.py files and the 'Frank_R', 'NISTQueryFiles' and 'TestingFiles' folders. However, the contents and functional significance of these additional components will be discussed in detail in the subsequent sections. To provide context, screen dumps of selected pages have been included within Appendix D.

4.3 LC-MS Data-Dependent Acquisition (MS/MS)

Relevant Use Cases: Derive peak data from uploaded sample files, View fragmentation spectra.

Upon completion of file upload, the 'Create Fragmentation Set' button is available to the user on the 'Experiment' page. Clicking the link renders a simple form, allowing the user to enter a unique name for the new fragmentation set. Submission of the form generates a POST request, which is directed via the url.py file to the *create_fragmentation_set* view. The view validates and processes the form, generating a corresponding FragmentationSet object. At this juncture, the *create_fragmentation_set* view performs an additional validation step to

ensure that at least one sample file has been uploaded to the experiment prior to the commencement of peak data extraction. If valid, the new FragmentationSet is committed to the database.

Following creation of the Fragmentation Set object, the method *input_peak_list_to_database* is called from within the view. This method determines the experimental protocol of the experiment, from which the peak data of the Fragmentation Set is to be derived, and ensures peak data extraction is performed in a manner appropriate to the protocol. The duration of the process of peak extraction can vary depending upon the volume of data to be processed. Therefore to prevent the application from becoming unresponsive, Celery was implemented to perform processes asynchronously. Celery is based on distributed message passing, which allows for the asynchronous processing of queued tasks by 'workers' (Celery Project, 2015). Upon the identification of the LC-MS/MS-DDA experimental protocol, the *input_peak_list_to_database* adds the *msn_generate_peak_list* task (located in the tasks.py file of the application) onto the queue for asynchronous processing.

As the name suggests, *msn_generate_peak_list* is the method responsible for the extraction of the peak data from the source files. The process of extracting peak data is complex. As such, the staff of the Metabolomics facility utilise existing R scripts to derive peak data. In order to generate a list of peaks from the mzXML files, the package rpy2 was used to provide a low-level interface from Python to R. The 'frankMSnPeakMatrix.R' script is sourced from the 'Frank_R' folder of the application and run by passing the root directory of the experiment.

The 'frankMSnPeakMatrix.R' script (provided by Joe Wandy, modestly adapted), extracts the peaks from each source file using the xcmsSet function of the XCMS package. The lineage of each product peak is then derived using the 'frankXcmsSetFragments.R' script (provided by Tony Lowson). The candidate MSN scans for each precursor peak are identified, and grouped. The selection criteria of which scan the product peaks are to be derived from can vary. However, at present the default is to derive the product peaks of the parent from the scan which corresponds to the highest precursor peak intensity. Once the lineage of the peaks is determined, the R subprocess returns a tabular list of the peaks which includes their msn level and any precursor peak. The 'peak list' is returned to the FrAnK application in the form of an rpy2.robjects data.frame.

Processing of the 'peak list' is performed by the MSNPeakBuilder class (located in peakFactories.py) which validates the rpy2 dataframe upon construction. The *populate_database_peaks* method of the class is called by the *msn_generate_peak_list* task to initiate the conversion of the dataframe peak list to Peak objects and their subsequent addition to the database. The algorithm for the population of peaks from the peak list is detailed in figure 8.

The algorithm is designed to ensure that only those peaks with an associated fragmentation spectrum are populated into the database. This ensures the redundancy of storing peaks with no fragmentation spectrum is prevented as these peaks cannot be queried against a spectral reference library to retrieve candidate annotations. Although recursive algorithms are typically resource intensive, the client specified in the initial brief that the maximum number of MSN levels typically ranges from 3-5 levels. As such, it is not envisaged that the implementation of a recursive algorithm would be deleterious to the performance

of the application. In addition, a dictionary was utilised to implement the algorithm. As the peak list is created in R, the 'peak ID' assigned to each peak in the dataframe has no relevance in the FrAnK application. As such, the creation of each peak includes its addition to a 'created peaks' dictionary, storing the R peak ID as the key and the primary key ID of the corresponding object in the application's database as the value. Therefore, the frequency of database look-up is diminished and the efficiency of the algorithm is improved.

Populate Database Peaks Algorithm

1. Set *start* to *n* peaks in peak list
2. For *peak* = *start*, …, 1 in peak list, repeat:
    I. If the peak has not been created previously and has a parent peak:
        i. Get the parent peak
        ii. Create the peak, adding the peak to the database
3. Terminate

Get the Parent Peak Algorithm

1. If the parent peak has been created previously, retrieve it from the database:
    1. Return the parent peak
2. Else, the parent peak has not been created :
    I. If the parent peak has a precursor peak:
        i. Get the parent peak
    II. Create the parent peak, adding the peak to the database
    III. Return the parent peak

Figure 8: Population of LCMS-MS/MS-DDA peak list algorithm

Upon completion of peak data extraction, the status of the Fragmentation Set object is changed from 'Processing' to 'Completed Successfully' allowing the user can access the fragmentation set page of the application. This page displays the MS1 peaks derived from the source files. Selection of an MS1 peak, navigates the user to the peak page which displays the product peaks which comprise the fragmentation spectra of the precursor.

4.4 GCMS Electron Impact Ionisation

Relevant Use Cases: Derive peak data from uploaded sample files, View fragmentation spectra.

As previously discussed in section 4.4, the creation of a fragmentation set by the user initiates the calling of the method *input_peak_list_to_database* by the *create_fragmentation_set* view. In contrast to the identification of LC-MS/MS-DDA, the identification of the GCMS-EII experimental protocol causes the initiation of the *gcms_generate_peak_list* task. The extraction of peak data from source files generated from the GCMS-EII experimental protocol follows a similar pattern to that of the LC-MS/MS-DDA described previously. However, as discussed in section 1.1.10, the analysis of GCMS fragmentation spectra is confounded by the absence of a measurable parent peak due to the harsh nature of electron impact ionisation. As such, a distinct R script was required to group the derived peaks into distinct fragmentation spectra corresponding to the parent analyte.

Populate Database Peaks Algorithm

1. For file in source file list, repeat:
    I. Group Peaks
    II. Add Peaks to Database

Group Peaks Algorithm

1. Open R output file
2. Let *current_group_id* = -1 and instantiate *grouped_peaks* dictionary
3. For line in output file, repeat:
    I. Extract *peak_mass*, *retention_time*, *intensity* and *peak_relation_id* from line
    II. Create peak object, setting *msn_level* =2 and *parent_peak*=None
    III. If *peak_relation_id* is not equal to *current_group_id:*
        i. Set *current_group_id* to *peak_relation_id*
        ii. Add *peak_relation_id* key to *grouped_peaks*
    IV. Add peak object to *grouped_peaks*[*peak_relation_id*]
4. Terminate yielding *grouped_peaks*

Add Peaks to Database Algorithm

1. For relation in *grouped_peaks*, repeat:
    I. Let *pseudo_ms1_peak* = *grouped_peaks*[*relation*][0]
    II. For peak in *grouped_peaks*[*relation*], repeat:
        i. If the intensity of the peak is greater than *pseudo_ms1_peak* intensity:
            a. pseudo_ms1_peak = peak
    III. Create the *pseudo_ms1_peak* object, assigning *msn_level* = 1
    IV. For *peak* in *grouped_peaks*[*relation*]
        i. Set *peak.parent_peak* = *psudo_ms1_peak*
        ii. Commit *peak* to the database

Figure 9: GCMS-EII Peak Population Algorithm

Using rpy2, the *gcms_generate_peak_list* task sources the 'gcmsGeneratePeakList.R' script located in the 'Frank_R' folder of the application. The 'gcmsGeneratePeakList.R' script is an adaptation of an existing R script provided by the client. The adaptation was required as the existing R script performs the grouping of GCMS-EII data across distinct source files, whereas each input file should be considered in isolation within FrAnK. In a similar manner to the 'frankMSnPeakMatrix.R' script implemented in the analysis of LCMS-MS/MS-DDA, the 'gcmsGeneratePeakList.R' uses the xcmsSet function of the XCMS package to derive peaks from the mzXML files. From which a PeakML file is written from the XCMS set. While the mzML and mzXML file formats store raw MS data, the PeakML file is an XML file format for the storage of extracted features (Scheltema *et al*., 2011). The peaks within the PeakML file are then filtered to remove noise, and those peaks whose intensity is below a set threshold. The mzMatch *ipeak.sort.RelatedPeaks* method is then used to group peaks to establish their relation. Finally, the peak list is output to a text file and the R script returns a dataframe to the FrAnK application consisting of two character vectors corresponding to the name of the input file and text output file of the R script.

Upon the returning of an R dataframe, the *gcms_generate_peak_list* task instantiates a GCMSPeakBuilder object which is passed both the R dataframe (represented in python as an rpy2 data.frame object) and the primary key of the fragmentation set to be populated. The constructor for the GCMSPeakBuilder performs validation of the R dataframe and the fragmentation set id. The *gcms_generate_peak_list* task then calls the *populate_database_peaks* method of the GCMSPeakBuilder instance to begin the population of the peak list into the database. A summary of the algorithm used by the *populate_database_peaks* method of the GCMSPeakBuilder is provided in figure 9.

In order to circumvent the challenge associated with the lack of a detected precursor peak, a pseudo-MS1 peak was introduced to maintain the lineage of parent and product ions. In each relation of peaks, the pseudo-MS1 peak corresponds to the duplication of the product ion which is most abundant. While this may seem redundant, the inclusion of the pseudo-MS1 peak is necessary to indicate that the remaining peaks in the relation are product ions despite the absence of a measurable precursor. Therefore, why not simply 'promote' the most abundant peak to the MS1 level? While this was considered, the pseudo-MS1 peak remains a component of the fragmentation spectrum and therefore must remain in the MS2 level for identification when querying a spectral reference library.

Although the algorithm successfully populates the GCMS-EII peaks into the database, subsequent iterations of the application should look to refactor the algorithm's implementation. Retrospectively, there is redundancy introduced by the failure to identify the most abundant peak during the grouping of peaks into a dictionary from the R text output file. At present, the *grouped_peaks* dictionary stores a list of the product peaks for each grouping of peaks (identified by the 'relation id' key). However, the resolution to this inefficiency would be to store a dictionary as opposed to a list. As such, each relation identified would store a list of the peaks in one key:value pair and the most abundant peak in a second key:value pair. This would remove the unnecessary additional iteration of all the peaks in the relation to identify the most abundant peak in the *add_peaks_to_database* component of the algorithm. Therefore, the efficiency of the implementation could be improved upon.

In a similar manner to process described in section 4.4, the completion of the peak extraction process is relayed to the user via the update of the Fragmentation Set object status to 'Completed Successfully'. This allows for the fragmentation set page to become accessible to the user and allows for inspection of the peak data.

4.5 MassBank Web-API

Relevant Use Cases: Generate or retrieve candidate annotations for MS peaks.

From the fragmentation set page of the application, the user may select an AnnotationTool for the retrieval of candidate annotations for the peaks associated with the FragmentationSet object. Upon clicking of the 'Create New Annotation Query' button, a POST request is submitted to the *fragmentation_set* view of the application. Upon validation of the form, the name of the annotation tool selected by the user is extracted from the form. As such, the user is

navigated to the define annotation query page of the application. Dependent upon the annotation tool selected and the experimental protocol associated with the experiment from which the FragmentationSet object was derived, a form containing the relevant fields for the tool and choices appropriate to the protocol are displayed to the user. Having reviewed the documentation and tutorials available for the batch querying services provided by numerous spectral reference libraries, it was apparent that the standard of documentation and help available to those novice users was generally poor. As such, it was indented that the tailoring of potential search parameters and choices in FrAnK would simplify the retrieval of candidate annotations for novice users.

Upon submission of the parameters for the annotation query, a POST request to the *define_annotation_query* view of the application derives the annotation tool from a slug contained in the request. Therefore, the form specific to the annotation tool is retrieved from the POST request. In order to store the parameters of the search to be performed, the *set_annotation_query_parameters* method located in views.py is called. Using the form to establish which AnnotationTool has been specified, and consequently which parameters should be stored, the method extracts the user's input and populates the *annotation_tool_params* field of the new AnnotationQuery object in the form of a serialised dictionary. The populated AnnotationQuery object is returned to the *define_annotation_query* view. Subsequently, the method *generate_annotations* is called, which determines the AnnotationTool to be queried from the AnnotationQuery object and calls the '*massbank_batch_search*' task (tasks.py) if the user has selected to query the MassBank Web-API. As such, the retrieval of candidate annotations for the peaks associated with the FragmentationSet object is performed as a background process using Celery.

The *massbank_batch_search* task instantiates a MassBankQueryTool (located in annotationTools.py) passing the IDs of both the FragmentationSet and AnnotationQuery objects to be processed for validation. Upon which, the method *get_mass_bank_annotations()* method of the instance is called by the *massbank_batch_search* task. The retrieval of candidate annotations from MassBank can be considered as three distinct processes. Initially the fragmentation spectra to be queried are formatted and then sent to the MassBank API in the form of a Simple Object Access Protocol (SOAP) request. Upon retrieval of the results from the API, the compounds and candidate annotations are populated into the database. A summary for the creation of the query spectra is provided in figure 10.

Although the MassBank Web-API allows for the submission of both positive and negative spectra simultaneously, it was decided that to improve the veracity of candidate annotations the spectra should be send as two distinct queries to the API. To facilitate the sending of SOAP requests to the MassBank Web API, the Suds web services client for python was implemented. The serialised search parameters are extracted from the AnnotationQuery object and are included within a dictionary containing the query spectra for submission to the API via the Suds client. The client will intermittently query the MassBank Web API to receive updates of the status of the submitted task. Upon completion, the results are returned by a final query to the Web API.

Generate Query Spectra Algorithm

1. Let *peak_set* be all the peaks within the fragmentation set
2. Let *num_levels* be the maximum *msn_level* in the *peak_set*
3. Instantiate a *positive* and *negative_query* spectra list
4. For *level* in range (1, *num_levels*):
    I. Let *peaks_in_level* equal the peaks in the *peak_set* with *msn_level* = *level*
    II. For *peak* in *peaks_in_level*:
        i. Retrieve the *fragmentation_spectrum* associated with the *peak*
        ii. If the length of the *fragmentation_spectrum* > 0:
            a) Add the identifier of the *peak* to the *query_list*
            b) For *fragment* in *fragmentation_spectrum*:
                A. Add the mass and intensity of the *fragment* to *query_list*
            c) If polarity of *peak* is positive:
                A. Add the *query _list* to the *positive_query* as a string
            d) Else if the *peak* is negative:
                A. Add the *query_list* to the *negative_query* as a string
5. Add the *negative_query* and *positive_query* lists to a *query_spectra* dictionary
6. Terminate returning *query_spectra*

Figure 10: Generate Query Algorithm for Massbank Batch Search

The candidate annotations are returned to the application in the form of a list, containing a dictionary corresponding to each of the submitted spectra. Each spectrum is related back to its associated peak by reference to the unique peak slug included in the query. As such, the compound and annotation data are parsed from the returned results and are populated to the database. Finally, the status of the AnnotationQuery object is updated to relay the completion of the process to the user. For each AnnotationQuery object submitted by the user, the returned candidate annotations are displayed in a tabular format via the peak page of the application.

4.6 NIST14

Relevant Use Cases: Generate or retrieve candidate annotations for MS peaks.

Upon the submission and processing of the NISTQueryForm and the generation a new AnnotationQuery object, the *generate_annotations* method located in views.py will call the *nist_batch_search* task. In a similar manner to the MassBankQueryTool, the instantiation of the NISTQueryTool validates the input of the ID of the AnnotationQuery object to be performed. The *nist_batch_search* task then calls the *get_nist_annotations* method of the NISTQueryTool object to begin the retrieve candidate annotations.

The NIST14 Mass Spectral Library is accessible locally, via the MS PepSearch software developed for Windows. MS PepSearch, queries a peak list provided within an MSP file format via command line arguments, against the various reference libraries of NIST14. However this posed a significant challenge to its implementation, as a requirement of FrAnK was to maintain compatibility with the existing hardware and software of the client's pipeline to ensure integration (*see section 2.8*). As the existing server runs on a Linux-like operating system, the compatibility layer software application Wine was implemented to allow for to be performed via the MS PepSearch program.

The initial step of the *get_nist_annotations* method of the NISTQueryTool object is to write a temporary MSP file to the 'NISTQueryFiles' folder of the application. The MSP file contains the individual fragmentation spectra associated with the FragmentationSet object to be annotated (*figure 11*). While similar to the algorithm used to format the queries sent to the MassBank API (*figure 10*), the algorithm is distinct due to the determination of whether the precursor peak's *m/z* is to be incorporated into the query. Its inclusion is unnecessary for queries to the MassBank Web API but is important when querying NIST14 libraries. The local installation of NIST14 comprises five distinct spectral reference libraries which correspond to distinct experimental protocols. The 'mainlib' corresponds to fragmentation spectra derived from GCMS-EII experiments whereas both the 'nist_msms' and 'nist_msms2' libraries consist of spectra obtained from LCMS/MS experiments. This distinction is highly relevant to the retrieval of candidate annotations from the 'mainlib', as the process of extraction of peak data from the GCMS source files generates a pseudo MS1 peak in the application (*see section 4.4*). As such, erroneous candidate annotations would be retrieved from the 'mainlib' of NIST14 were the precursor *m/z* included in any query submitted.

Generate Query Spectra Algorithm

1. With *query_file* open:
    I. Let *peak_set* equal all the Peak objects associated with the FragmentationSet object to be annotated.
    II. Let *num_levels* be the maximum *msn_level* in the *peak_set*
    III. For *level* in range (1, *num_levels*):
        i. Let *peaks_in_level* equal the peaks in the *peak_set* with *msn_level* = *level*
        ii. For *peak* in *peaks_in_level*:
            a) Retrieve the *fragmentation_spectrum* associated with the *peak*
            b) If the length of the *fragmentation_spectr*um> 0:
                A. Write the details of the *peak* to the *query_file.* Include the precursor *m/z* only if the experiment is LCMS and not GCMS.
                B. For *fragment* in *fragmentation_spectrum*:
                    a. Add the mass and intensity of the *fragment* to *query_file.*
2. Finally, ensure the *query_file* is closed.

Figure 11: Generate Query Algorithm for NIST14 Batch Search

Upon completion of the writing of fragmentation spectra to the temporary MSP file, the *get_nist_annotations* method proceeds to formatting the call to MS PepSearch. This ensures the retrieval of candidate annotations are performed in accordance with the search parameters provided by the user via the NISTQueryForm. The user-specified parameters include the maximum number of hits to be retrieved for each spectrum, the search type and the reference libraries to be queried. In addition, the call also includes parameters which are associated with the NIST AnnotationTool object itself. Namely, the location of the MS PepSearch program and the NIST libraries within the directories reserved for the Wine program are derived from the AnnotationTool instance itself in the form of a serialized dictionary. The default parameters of the NIST AnnotationTool object can be altered via the population script populate_pimp.py.

The NIST14 libraries are then queried through the subprocess call to Wine, which outputs the candidate annotations as an additional temporary file within the 'NISTQueryFiles' folder. The output file is then read by the FrAnK application. The candidate annotations are then parsed, and populated into the database. Finally, the temporary files stored in the 'NISTQueryFiles' folder are removed and the status of the Annotation Query object is set to 'Completed Successfully' to indicate to the user the completion of the task.

4.7 Testing Strategy

Within the tests.py file of the application, tests have been provided for the FrAnK application. A summary of the end-to-end test cases implemented are shown in Appendix D. While complete unit testing was desirable, this could not be achieved in the time provided. As such, the testing strategy implemented was to prioritise end-to-end tests to encompass the testing of the views, tasks and models. Each of the views within the application has been tested to ensure the page renders successfully and those which include forms have been tested with varying inputs.

One of the challenges of testing the FrAnK application was the use of Celery. In order to include the celery-performed tasks within the end-to-end testing, celery provides a 'TEST-RUNNER' variable within the 'settings_dev.py' file of the project folder. This allows for the typically asynchronous processes contained within the 'tasks.py' file of the application to be performed in a synchronous manner during testing. Therefore, the end-to-end tests encompass both the testing of the views and background processes. However, it is acknowledged that this is not the ideal practice. The 'TestingFiles' folder of the application contains sample GCMS and LCMS data files which are uploaded during the running of tests. It should be noted that the duration of time required to perform the tests provided may be extensive due to the volume of data contained within these files. As such, these tests have been commented out to allow for rapid testing. Nevertheless, this was a preferred approach as it was considered that testing aught to be performed using realistic datasets. As such, the sample mzXML files provided within the 'TestingFiles' folder are representative MS datasets which were provided by the client.

As shown in Appendix D, the end-to-end testing demonstrates the functionality of the application. However, additional tests have been included in the 'tests.py' file and additional tests could be implemented to improve the coverage of the code. Notably, two tests were not passed by the current implementation. The first, which was to test the MassBank Web API, could not be performed due to the availability of the service. However, the second failure was derived from the failure of the *specify_preferred_annotation* view. Upon failure of the test, this feature was investigated and appeared to be working as anticipated. As such, the failure is likely due to the Test case as opposed to the implementation.

Chapter 5 Evaluation

5.1.1 Client Evaluation

The evaluation of the FrAnK application was performed alongside the client, Dr Karl Burgess, head of the Metabolomics Facility at Glasgow Polyomics. The evaluation consisted of two distinct elements. Initially the client was asked to perform a series of simple tasks to demonstrate the product and to garner feedback from the perspective of the user. In addition, the client was asked to a series of open-ended questions to garner further insight into the application, the development process and the client's view of the future development of the application. Further details of the format of the client evaluation are provided in Appendix E.

When asked for a general impression of the development of the FrAnK application, the client stated that the development had gone fantastically well that the process of evaluating prototypes at regular intervals was extremely beneficial. Subsequently, when asked if the existing functionality of FrAnK would be of value to the research staff of the facility the client confirmed that this would indeed be the case. Elaborating further, the client explained that researchers within the field of metabolomics utilise fragmentation support on a frequent basis. As such, the client anticipates that within approximately 6 months it will be considered unacceptable to publish work without supporting evidence provided through fragmentation analysis. Therefore, the implementation of fragmentation support within the existing analytical pipeline is of the upmost importance to the research staff within the facility. A secondary consideration to the client was that commercial software is emerging to market which supports fragmentation analysis. However, the available open-source products have, thus far, failed to provide support. As such, the development of the application represents a competitive advantage within the field.

One of the novel features of the application is that the framework readily supports the inclusion of additional spectral reference libraries. As noted in section 2.5, the existing competing applications provide limited coverage of the metabolome due to the sparse selection of reference libraries supported. As such, the client expressed that a key motivation for the development of FrAnK was that a broad range of libraries could be queried from within a single application. The client explained that this would be considered a key feature of the application, as chemicals must either be purchased or synthesized in order for a reference fragmentation spectrum to be determined. As such, each spectral reference library consists of distinct compounds. When analysing a biological sample, it is often insufficient to simply query a single point of reference as rare metabolites are often a source of scientific interest. As such, the querying of a single library may not yield suitable candidate annotations if the metabolite is absent from the reference library. Therefore, the querying of multiple reference databases will become integral to generating scientific publications.

The interview proceeded to discussion of the user interface. At this juncture, the client expressed that the use of the existing PiMP base template was a positive to improve familiarity for the user. However, when I enquired further, the client

confirmed that additional descriptive text should be added to provide clarity to the application. The client indicated that the addition of a 'Wizard-like' feature would be beneficial or pop up help text. However, this could not be implemented in the development time remaining. As a compromise, additional help text was provided to each of the form fields and into the HTML pages to try to clarify any existing points of potential confusion. It was then posed to the client whether the data stored within the application was appropriate in style and content to allow for the researcher to evaluate candidate annotations effectively. The client responded that the application conveyed most of the required data; however, the client enquired as to the feasible to display the fragmentation spectrum of the candidate annotation to allow for a visual comparison between the measured spectrum and that of the candidate. Whilst the confidence value returned by the spectral reference libraries is a quantification of the match between spectra, the client explained that a visual comparison provides valuable insight to the researchers. The client has proposed that this may take the form of a 'mirror-like' display, consisting of the measured spectrum above and the inverted candidate spectrum underneath. Therefore, peaks within the spectra that do not align by m/z can readily be identified.

As the client had successfully completed the tasks provided to demonstrate the application, when asked if there were any points during the evaluation task which were unclear or challenging the client simply responded that there were not and the tasks were straightforward. In addition, the client was asked whether or not the development had achieved the outcomes envisaged. The client responded that the anticipated outcomes were "pretty much" achieved, clarifying that with the exception of integration with the existing application PiMP, the application behaves as anticipated.

Finally, the evaluation turned towards which features the client would wish to be implemented in the future. The client responded that there would be many features that would be of use to add into the existing FrAnK application. The obvious next stage of development would be to integrate the FrAnK application with PiMP in a manner conducive to supporting fragmentation analysis in the main pipeline. In addition, the client emphasised that additional spectral reference libraries would be added to improve coverage of the metabolome. Finally, it would be useful to consolidate the confidence scores retrieved across spectral reference libraries, generating a single representative confidence score.

5.1.2 Qualitative Evaluation of Standards

As stated previously, the aim of the application was to assist in the identification of metabolites of interest in biological samples. As such, the testing of the application using data derived from biological samples poses a challenge as their chemical composition is unknown. Therefore, a qualitative analysis of the application was performed using MS data files, corresponding to standard solutions, provided by the client. In MS, a standard is a solution created in the laboratory and as such, its constituents are known. While peaks derived from standards are commonly used to aid in the identification of metabolites within biological samples, they have been used to determine the veracity of the candidate annotations retrieved by the FrAnK application in this context.

As such, peaks were extracted from the standard files and the candidate annotations retrieved through the querying of the NIST14 'nist_msms' and

'nist_msms2' libraries were evaluated. Unfortunately, equivalent candidate annotations could not be assessed from the MassBank Web API as, at the time of evaluation, the batch service was 'temporarily suspended' (17/08/2015). The table shown in Appendix E summarises a sample of the compounds, on the left of the table, known to be included in the standards mixtures. On the right of the table, the values of a peak and an associated candidate annotation retrieved by NIST are shown. While these results are to be interpreted parsimoniously, as the peaks selected are not representative of the performance of the application as a whole across all peaks and are admittedly 'cherry-picked', they nevertheless appear to suggest that the application is performing as anticipated. In the examples provided, the candidate annotation with the greatest confidence is typically the mass of a proton different from the molecular mass of the compound. Furthermore, in instances in which the candidate annotation does not appear to match the compound included in the standard mixture, a similar molecular formula is retrieved or a clear derivative of the standard compound has been identified.

While these results do not confirm that the application performs as intended, they are suggestive that plausible candidate annotations have been retrieved by the FrAnK application from the spectral reference library.

Chapter 6 Discussion and Conclusion

6.1 Development Challenges

Throughout the development of FrAnK, numerous obstacles had to be overcome. Despite a scientific background, the depth of the domain knowledge required to comprehend the requirements of the application, generate design documentation and provide viable solutions during the implementation was extensive and at times overwhelming. The project supervisors and staff at the Metabolomics Facility provided extensive support and were extremely patient throughout. Through the sourcing of relevant literature and constant quizzing of domain experts, I believe the challenges associated with my initial lack of domain knowledge were largely overcome.

Nevertheless, the development of FrAnK posed significant technical challenges throughout. Foremost of these was the integration of such a range of diverse technologies, which were necessitated by the requirement to ensure the application maintained compatibility with the existing pipeline. From the onset, significant time was required to install the existing PiMP application, study its database schema and processes to gain insight into the analytical pipeline. In particular, PiMP integrates Django, MySQL, R (underpinned with Java) and Celery. Furthermore, it has been previously acknowledged that the R scripts used within FrAnK to extract peak data from LC-MS/MS data files were kindly provided. However, this in itself proved to be a challenge as, having no prior experience in the language, it was necessary to comprehend the complex processes contained within to appreciate the significance of inputs and outputs.

While these challenges were faced and overcome during the initial evaluation phase of the project, the process of implementing the functionality required by FrAnK posed unique challenges. For the process of peak extraction from GCMS-EII data sets I was provided with an R script which generates a peak list derived across numerous source files. However, this was unsuitable for the FrAnK application, which was to treat each source file in isolation. As such, I was required to generate a distinct R script for the generation of a GCMS-EII peak list. In addition to volume of data to be processed, the application's use of Celery posed significant challenges due to the necessity to manage concurrent database transactions. Furthermore, in order to achieve the retrieval of candidate annotations it was required that Suds, Wine and MS PepSearch be implemented in the FrAnK application.

While both the challenges associated with the domain and technologies can be considered distinct, they were at times intertwined. When sourcing suitable spectral reference libraries, I was required to investigate the documentation provided for each API. In addition to the diversity of technologies used, the overall clarity of the documentation was poor, with both technical and domain concepts explained in insufficient detail. The final challenge associated with the retrieval of candidate annotations was that the format of chemical data varies between spectral reference libraries. As such, the candidate annotations output by each API vary in both format and content.

6.2 Limitations of Existing Application

At present, the FrAnK application lacks the ability for the user to modify, delete and share existing experiments, fragmentation sets and annotation queries. Furthermore, the aesthetic appearance of the application requires additional development. In response to the client evaluation, attempts were made to include additional help text to ease the process of learning for new users of the application. Nevertheless, it would be of great benefit for future iterations of the application to provide either a help Wizard or a tutorial feature. Despite significant effort, it is acknowledged that these limitations limit usability.

While FrAnK prevents the user from uploading files which are not in the mzXML file format, additional checks could be included to minimize user errors. Using an XML parser, the files could be checked to ensure that product peaks are included in the files. Furthermore, the addition of the XML parser could be used to determine the polarity of the file from the source as it is performed in PiMP. However, at present, the user is required to specify the polarity of the file within the page form. In addition, the current application cannot differentiate between files derived from the LCMS and GCMS experimental protocols. This would be highly beneficial in avoiding user error. However, it is unclear at present how such a check could be implemented as MS instrumentation which derives the file may not be aware of the sample preparation processes which precede the analysis.

With the retrieval of candidate annotations from both MassBank and NIST, there is potential for irregular variants in the format or content of the annotations returned which may cause errors in the application. For example, the name of the compound may exceed the maximum size of the CharField used to store the name of the chemical compound. Furthermore, as a public repository the data content and format of the candidate annotations returned from MassBank vary significantly. To date, all files supplied by the client have been annotated using NIST. However, it was not possible to annotate the peak data for the GCMS data files using MassBank, due to the suspension of the service. An additional limitation to be acknowledged is that the database schema recognizes that a given compound may originate from numerous spectral reference libraries. However due to the lack of standardization in compound naming conventions and identifying codes, the same compound may be erroneously duplicated in the FrAnK database at present if it is retrieved from distinct spectral reference libraries.

Finally, an additional limitation of the application is the use of Matplotlib to generate the graphical display of the fragmentation spectra. The visualization of the fragmentation spectra should be implemented using Highcharts as is implemented within the PiMP application. As such, visualization could be made interactive to allow for navigation through the MSn levels of the data.

6.3 Future Work

The primary focus of future development upon FrAnK should focus initially upon the limitations identified in the previous section. However, upon completion the application could then be integrated into the pipeline to supplement the existing functionality within PiMP. Nevertheless the client has described within the

evaluation that fragmentation support is essential going forward. Due to the similarities between the two databases, integration could be achieved in the form of an additional task added to the tasks.py file. Taking a PiMP project as input, the task could generate the corresponding experiment entry in the FrAnK database. From which peaks and, subsequently, candidate annotations could be derived from the fragmentation spectra. The candidate annotations could then be presented in the PiMP application to allow for metabolite identification. Although this is likely to introduce redundancy into the data, it is nevertheless required in order for FrAnK to maintain standalone functionality.

While candidate annotations can be retrieved from the spectral reference libraries MassBank and NIST from within the FrAnK application, the true potential of the application can be realised with the addition of bespoke software and algorithms in the form of additional annotation tools. The candidate annotations derived from the spectral reference libraries may provide input for *post hoc* manipulations to improve the veracity of the annotations. While valid, the querying of mass spectral libraries alone is somewhat limited due to the dependency upon the coverage of the metabolome of the library. As such, many metabolites may go unidentified due to the lack of coverage offered by the existing services. However, the inclusion of additional spectral reference libraries would diminish this risk. During the client brief additional libraries were commented upon as suitable candidates and their addition would significantly improve the coverage of the metabolome included within the application, leading to greater reliability in metabolite identification as numerous sources could be used to provide supporting evidence.

Subsequent iterations of FrAnK could include additional experimental protocols such as the LCMS-MS/MS data-independent acquisition, which could not implemented during the current project. In addition, additional parameters for the peak extraction tasks, which generate Fragmentation Sets, could be implemented to allow for greater flexibility. While the R scripts provided have been parameterized over the course of the project, the application does not currently include a form to support this. However, this should be considered a low priority as the client has stated previously that the workflow should remain as simply as possible for the research staff. As such, the additional parameters could be implemented as 'advanced settings'. Furthermore, it would be of merit for the user to be able to specify the source of the MS1 peaks to which MS/MS fragments are associated. These could be derived from a user uploaded peak list, a distinct mzXML file or from the existing PiMP application. These features would be of benefit to the identification of metabolites as the process of fragmentation reduces the frequency with which MS1 peaks can be measured by the instrumentation, thereby diminishing the resolution of the full-scan. At present, an existing R script has been developed by Joe Wandy which could be incorporated into FrAnK, replacing the existing LCMS/MS R script, to achieve this aim.

As stated in the section 6.1, the lack of standardization within MS data and the format in which it is stored is noteworthy. As such, the current compound table within FrAnK will become unwieldy as new annotation tools are added into the application. Redundancy could be reduced by the querying of the returned annotations against the Chemical Translation Service's master list of compounds. This service stores a catalogue of the repository-specific compound identifiers to allow for comparison between spectral reference libraries. At

present, PiMP implements this to maintain standardization within its compound table of the database and this should be extended to FrAnK. While a challenging endeavor, this would certainly be advantageous going forward.

6.4 Conclusions

With the recognition that the constituent metabolites which comprise biological systems cannot be investigated in isolation, it is inevitable that untargeted approaches will supersede conventional targeted studies. While untargeted approaches have gained traction in disciplines such proteomics and genomics, the software tools and algorithms necessary to realise the potential of metabolomics remain to be developed. In particular, the existing software to identify candidate annotations for MS peaks is limited.

The FrAnK application has been developed to meet the need of improving metabolite identification through the utilization of fragmentation spectra. In the short-term, FrAnK provides the research staff at the Metabolomics Facility with a tool to significantly increase the rate at which MS data may be analysed and interpreted. It is hoped that this may provide supporting evidence for future scientific publications and improve the efficiency of existing services. The inclusion of the peak extraction from GCMS-EII experiments expands the number of experimental protocols which are supported within the existing pipeline. Furthermore, the potential coverage of the metabolome within the existing application has been increased by the inclusion of both the MassBank and NIST spectral reference libraries.

The present application has limitations, which have been acknowledged in section 6.2. However, it is hoped that FrAnK may provide a framework, upon which, the much needed software and algorithms required to improve metabolite identification may be implemented and integrated. The veracity of the candidate annotations will only increase as additional spectral reference libraries and bespoke annotation tools are added to the existing application.

Chapter 7 References

Allen F, Pon A, Wilson M, Greiner R and Wishart D (2014). CFM-ID: a web server for annotation, spectrum prediction and metabolite identification from tandem mass spectra. *Nucleic Acids Research*. 42: W94-95.

Azzopardi L and Maxwell D. How to Tango With Django 1.7. Url: http://www.tangowithdjango.com/book17/. Accessed 15/07/2015.

Berdie Rabanaque B, Casala i Ribes I, Fernandez Vidal I, Jauregui Pallares O, Marimon Corbella, RM, Perona Moreno J and Teixidor Casamitjana P (2012). Basics of Mass Spectrometry. *Handbook of instrumental techniques from CCiTUB*. http://hdl.handle.net/2445/32138.

Celery Project, 2015. Celery: Distributed Task Queue. Url: http://www.celeryproject.org/. Accessed 05/08/2015.

Chapman JD, Goodlett DR, Masselon CD (2014). Multiplexed and Data-Independent Tandem Mass Spectrometry for Global Proteome Profiling. *Mass Spectrometry Reviews*. 33: pp 452-470

Courant F, Antignac J-P, Dervilly-Pinel G, Le Bizec B (2014). Basics of Mass Spectrometry Based Metabolomics. *Proteomics*. **14**: pp 2369-2388.

Django Software Foundation, 2015. Django Documentation. Url: https://docs.djangoproject.com/en/1.7/. Accessed 20/07/2015

Egertson JD, MacLean B, Johnson R, Xuan Y, MacCoss MJ (2015). Multiplexed peptide analysis using data-independent acquisition and Skyline. *Nature Protocols*. **10**: pp 887-903.

Glish GL, Vachet RW (2003). The Basics of Mass Spectrometry in the Twenty First Century. *Nature Reviews Drug Discovery*. **2**:2 pp 140-150

Go EP (2010). Database resources in metabolomics: an overview. *Journal of Neuroimmune Pharmacology*. **5**: pp 18-30.

Heinonen M, Shen H, Zamboni N, Rousu J (2012). Metabolite identification and molecular fingerprint prediction through machine learning. Bioinformatics. **28**: pp 2333-2341.

Kessler N, Walter F, Persicke M, Albaum SP, Kalinowski J, Goesmann A, Niehaus K and Nattkemper TW (2014). ALLocator: an interactive web platform for the analysis of metabolomics LC-ESI-MS datasets, enabling semi-automated, user-revised compound annotation and mass isotopomer ratio analysis. *PLoS One*. **9**: e113909.

Mann M, Hendrickson RC, Pandey A (2001). Analysis of Proteins and Proteomes By Mass Spectrometry. *Annual Review of Biochemistry*. **70**: pp 437-73.

Marchetti AA and Mignerey AC (1993). Deconvolution of mass spectra. Nuclear Instruments and Methods in Physics Research. **324**: pp 288-296.

Marquet P, Saint-Marcoux F, Gamble TN, Leblanc JCY (2003). Comparison of a preliminary procedure for the general unknown screening of drugs and toxic compounds using a quadrupole-linear ion-trap mass spectrometer with a liquid chromatography-mass spectrometry reference technique. *Journal of Chromatography B*. **783**: pp 9-18.

mzMatch, 2015. Installation of mzmatch.R and mzmatch packages. Url: http://mzmatch.sourceforge.net/installation.php. Accessed: 05/08/2015.

Pivotal, 2015. RabbitMQ. Url: http://www.rabbitmq.com/. Accessed 05/08/15.

Scheltema RA, Jankevics A, Jansen RC, Swertz MA, Breitling R (2011). PeakML/mzMatch: A File Format, Java Library, R Library, and Tool-Chain for Mass Spectrometry Data Analysis. *Analytical Chemistry*. **83**: pp 2786-2793.

Smith R, Mathis AD, Ventura D, Prince JT (2014). Proteomics, lipidomics, metabolomics: a mass spectrometry tutorial from a computer scientist's point of view. *BMC BioInformatics*. **15**:S9.

Xia J and Wishart DS (2009). Web-based inference of biological patterns, function and pathways from metabolomics data using MetaboAnalyst. *Nature Protocols*. **6**: pp 743-760

Appendix A Glossary of Domain Terminology

"data-dependent acquisition" (DDA)

refers to the automated process of peak selection for dissociation based upon a criteria implemented following detection.

"data-independent acquisition" (DIA):

refers to the automated process of peak selection for dissociation based upon a criteria determined in advance of detection.

"electron impact ionisation" (EII):

a relatively 'harsh' ionisation technique, typically associated with gas chromatography, used prior to the analysis of samples through MS.

"electrospray ionisation" (ESI):

a relatively 'soft' ionisation technique, typically associated with liquid chromatography, used prior to the analysis of samples through MS.

"fragmentation spectrum":

the collective ions generated by the cleavage of a parent ion through dissociation.

"gas chromatography" (GC):

a technique which uses a gas mobile phase for the separation of analytes based upon their physical or chemical properties.

"parent ion" (GC):

refers to an ion which is dissociated to form fragments.

"product ion" (GC):

refers to a single ion generated from the cleavage of a parent ion.

"liquid chromatography" (GC):

a technique which uses a liquid mobile phase for the separation of analytes based upon their physical or chemical properties.

Appendix B Requirements Documentation

Introduction

The domain is metabolomics and mass spectrometry.

The University of Glasgow Metabolomics Facility implements a web-based pipeline, PiMP, to store and analyse mass spectrometry (MS) datasets. MS detects ions derived from the sample analytes which, in turn, can be cleaved to generate fragments which correspond to a substructure of the precursor. The identification of the compound from which an ion originated can be achieved through the querying of its associated fragmentation spectra to a spectral reference database. The current web application supports the analysis of peaks based upon the full scan of a sample (MS1). While the analysis of MS1 data alone can lead to the identification of potential candidate annotations, the validity of identifications can be significantly improved by incorporating the fragmentation spectra. The current system is unable to store and utilise this valuable structural information at present. Therefore an additional application is required to incorporate the analysis of fragmentation data to improve the identification of metabolites.

Summary of Client Brief

The aim is to provide additional functionality, prioritising peak identification as opposed to quantification, to the existing proprietary web-based pipeline PiMP. As such, the client has proposed the development of a new application capable of integration. However, the new application will be used independently of PiMP for the analysis of experimental protocols not currently supported by the pipeline. This additional functionality is to be achieved by incorporating the ability to store and utilise fragmentation spectra, ensuring the hierarchical relationship between parent and product ions is maintained. In addition, the data held for each peak should then be used to query a range of reference databases to identify candidate compounds. The identified candidates will be returned with a confidence score, which should also be stored in the database. The existing application relies upon R scripts to derive peak data from the source files. However, the functionality of the new application could deviate from the reliance upon R if necessary. In addition, the client has suggested that the ProteoWizard tool msconvert could be alternative solution to deriving the peaks from the source files. Furthermore, the client described an existing R script which has been developed to derive peaks, maintaining their relations, from datasets originating from data-independent acquisition experiments. However, this was written in haste and may require refactoring. The client has listed the following reference libraries which are of interest: NIST LCMS and GCMS libraries, Lipidmaps, MetFrag, MassBank, Magma/mzCloud and MSFrag. The client identified, and detailed, four scenarios in which fragmentation spectra should be utilised to retrieve candidate annotations.

Scenario 1: LCMS Data Dependent Acquisition (MS2)

- Full scan of the sample is performed, generating the MS1 data
- Peaks within a specific m/z range are selected based upon an experimenter defined criteria (such as top 10 most intense peaks).
- These peaks are subsequently dissociated to generate fragmentation spectra which are can be associated with the precursor peak.

- Multiple scans can generate duplication of peaks, due to an analyte continuing to elute over multiple time points. To avoid large quantities of redundant data from repeatedly fragmenting the same analyte, ions of a specific m/z can be added to an exempt list for a specified duration of time.

Scenario 2: LCMS Data Dependent Acquisition (MSN)

- Full scan is performed, generating the MS1 data
- Peaks are selected as previously described in *Scenario 1.*
- However, the peaks constituting the MS2 fragmentation spectra can themselves be selected and fragmented again. In theory, this process can be repeated n times. However, the client acknowledged that the maximum number of levels performed rarely exceeds 3, and at most 5.

Scenario 3: GCMS Electron Impact Ionisation

- Electron Impact Ionisation is a form of "hard" ionisation, unlike the Electrospray Ionisation commonly used for LCMS experiments.
- Electrons are focused into a beam, ionising the neutral molecules of the analytes but produces fragmentation of the analytes.
- Generates a fragmentation spectrum, however, the parent ion is missing.

Scenario 4: LCMS Data Independent Acquisition

- A technique which performs fragmentation of all ions within a specific m/z ratio range, which are above background noise levels.
- The technique consists of two scans, one at "high" energy (inducing dissociation) and a second at "low" energy.
- However, the product ions can still be associated with a parent ion

Documentation of Requirements

Functional Requirements

1. The application must enable authorised staff to create and edit experiments.
2. The application must enable authorised staff to create and edit experimental samples.
3. The application must enable authorised staff to upload data files to samples.
4. The application must enable authorised staff to derive peak data from uploaded data files.
5. The application must store the m/z, retention time and intensity of a peak.
6. The application must store the lineage of a peak.
7. The application must enable authorised staff to generate candidate annotations for a peak.
8. The application must store a confidence value for a candidate annotation.

9. The application must store the name, formula and mass of a compound identified by a candidate annotation.
10. The application must enable authorised staff to submit fragmentation spectra to a spectral reference library.
11. The application must provide a graphical representation of the fragmentation spectra for a peak.
12. The application must extract peak data from data files uploaded to PiMP.
13. The application must provide candidate annotations for analyses performed in PiMP.
14. The application should enable authorised staff to create and edit experimental conditions.
15. The application should enable authorised staff to specify a preferred annotation for a given peak.
16. The application should provide drag and drop file upload.
17. The application should allow authorised users to grant access to an experiment to other authorised users.
18. The application could display the chemical structure of a candidate annotation.
19. The application could display the fragmentation spectra of a peak alongside that of a candidate annotation for direct visual comparison.
20. The application could group peak data across experimental replicate source files.
21. The application would enable the exporting of peak data and candidate annotations.
22. The application would provide a tutorial for inexperienced users, unfamiliar with domain terminology.

Quality Requirements

1. The application must be capable of future enhancement to support the addition of novel experimental protocols and methods of generating candidate annotations.

Platform Requirements

1. The application must be developed using the Django framework (version 1.7)
2. The application must be compatible with client's existing server and database.
3. The application must support the mzXML file format.
4. The application could support the mzML file format.

Process Requirements

1. Prototypes of the application should be demonstrated at regular intervals.
2. Requirements for the application should be re-assessed and prioritised following each demonstration of a prototype.
3. The system should be delivered not later than 07/08/2015.

Key Use Case Descriptions

Experiment Creation

Name:	Create New Experiment	
Users:	Scientific Researcher	
Goals:	To create a new experiment for fragmentation analysis.	
Summary:	The user creates a new experiment, providing a description, title, experimental protocol and ionisation type.	
Priority:	MUST HAVE	
Preconditions:	The user must be authenticated, by having logged into the application.	
Steps:	User Actions	System Response
	1. Click on 'My Experiments' link.	
		2. Render 'My Experiments' page.
	3. Click "Add new experiment" link.	
		4. Render "Create Experiment" page.
	5. Input the name of the experiment, a description, the ionisation and experimental method and click "submit".	
		6. Process the form, and add experiment to database .
		7. Render 'My Experiments' page.
	8. Click on link to newly created experiment.	
		9. Render experiment page.
Post conditions:	A new experiment, associated with the user, has been defined in the database.	
Related Use Cases:	None	

Experimental Condition Creation

Name:	Create New Experimental Condition
Users:	Scientific Researcher
Goals:	To create a new experimental condition to an experiment.
Summary:	The user can add experimental conditions within an experiment by specifying a name and a description. This creates a logical grouping for experimental samples.
Priority:	SHOULD HAVE
Preconditions:	The user must be authenticated and have created the experiment

	previously.	
Steps:	User Actions	System Response
	1. Click 'Add Experimental Condition' link.	
		2. Render 'Add experimental condition' page.
	3. Input the name and a description for the experimental condition and click submit.	
		4. Process the form and add experimental condition to database.
		5. Render the experiment page.
	6. Click on the new experimental condition link.	
		7. Render experimental condition page.
Post conditions:	A new experimental condition, associated with an experiment, has been defined in the database.	
Related Use Cases:	Create New Experiment	

Experimental Sample Creation

Name:	Add Sample	
Users:	Scientific Researcher	
Goals:	To add a new experimental sample to an experimental condition.	
Summary:	The user can add samples to an existing experimental condition. The user inputs a name, description and an organism for a sample.	
Priority:	MUST HAVE	
Preconditions:	The user must be authenticated, having created an experimental condition within an existing experiment.	
Steps:	User Actions	System Response
	1. Click 'Add New Sample' link.	
		2. Render 'Add New Sample' page.
	3. Add a name, a description and state the organism of the sample and click submit.	
		4. Process the form and add sample to database.
		5. Render the experimental condition page.

Post conditions:	A new sample, associated with an experimental condition, has been defined in the database.
Related Use Cases:	Create New Experiment; Create New Experimental Condition

Data File Upload

Name:	Upload Data File	
Users:	Scientific Researcher	
Goals:	To upload a new data file to an experimental sample.	
Summary:	The user can upload data files via a form, providing the filepath of the file to be uploaded and selecting the polarity associated with the data file.	
Priority:	MUST HAVE	
Preconditions:	The user must be authenticated. The user must also create an experiment, defining the experimental conditions and samples.	
Steps:	User Actions	System Response
	1. Click 'Add New Sample File' link. 3. Specify the polarity and the filepath of the file to be uploaded and click submit.	2. Render 'Add New Sample File' page. 4. Process the form and create a sample file in the database. Upon which, the file is uploaded to an appropriate directory in the application. 5. Render the experimental condition page.
Post conditions:	A new data file, associated with an experimental sample, has been defined in the database and uploaded to the appropriate directory.	
Related Use Cases:	Create New Experiment; Create New Experimental Condition; Create Experimental Sample	

Extract Peak Data

Name:	Extract Peak Data
Users:	Scientific Researcher
Goals:	To extract the peak data from the uploaded sample files.
Summary:	The user can extract peak data by creating a fragmentation set, specifying the name of the set and providing any required parameters. Upon completion of peak data processing, the peaks data is added to the database maintaining their lineage.

Priority:	MUST HAVE	
Preconditions:	The user must be authenticated. In addition, the user must create an experiment (including conditions and samples) and have uploaded the requisite data files.	
Steps:	User Actions	System Response
	1. Click 'Create Fragmentation Set' link.	
		2. Render 'Create Fragmentation Set' page.
	3. Specify the name of the set, and any additional peak extraction parameters and click submit.	
		4. Process the form and create a fragmentation set in the database.
		5. Extract the peak data from the source files in the experiment associated with the fragmentation set.
		6. Render experiment page, displaying status of peak extraction process.
		7. Upon completion of step 5, update status of fragmentation set to 'Completed Successfully'.
	8. User clicks on the new fragmentation set link.	
		9. Render the fragmentation set page.
Post conditions:	A new fragmentation set, associated with the experiment, is created in the database. Furthermore, the peak data is populated in the database, associated with the fragmentation set, in a hierarchical manner.	
Related Use Cases:	Create New Experiment; Create New Experimental Condition; Create Experimental Sample; Upload Data File	

Generate Candidate Annotations

Name:	Generate Candidate Annotations for Peaks
Users:	Scientific Researcher
Goals:	To identify candidate annotations for each peak contained within a fragmentation set.
Summary:	The user selects an annotation tool, providing the tool specific parameters. The candidate annotations are then retrieved/generated

	and populated into the database for each peak in the fragmentation set.	
Priority:	MUST HAVE	
Preconditions:	User has successfully created a fragmentation set containing peak data in the database.	
Steps:	User Actions	System Response
	1. Selecting an annotation tool, the user clicks 'Create Annotation Query'.	
		2. Render 'Create Annotation Query' page, containing the form specific to the annotation tool.
	3. A name and tool-specific parameters are input and submit is clicked.	
		4. Process the form and add the annotation query to the database.
		5. Creation of the annotation query begins the generation or retrieval of candidate annotations from the annotation tool.
		6. Render fragmentation set page, displaying status of query process.
		7. Upon completion of step 5, update status of the annotation query to 'Completed Successfully'.
	8. Click on a link for a specific peak.	
		9. Render peak page, displaying candidate annotations in a tabular form.
Post conditions:	An annotation query is added to the database. Candidate annotations, and their associated compounds, are added to the database.	
Related Use Cases:	Extract Peak Data	

Select A Preferred Candidate Annotation

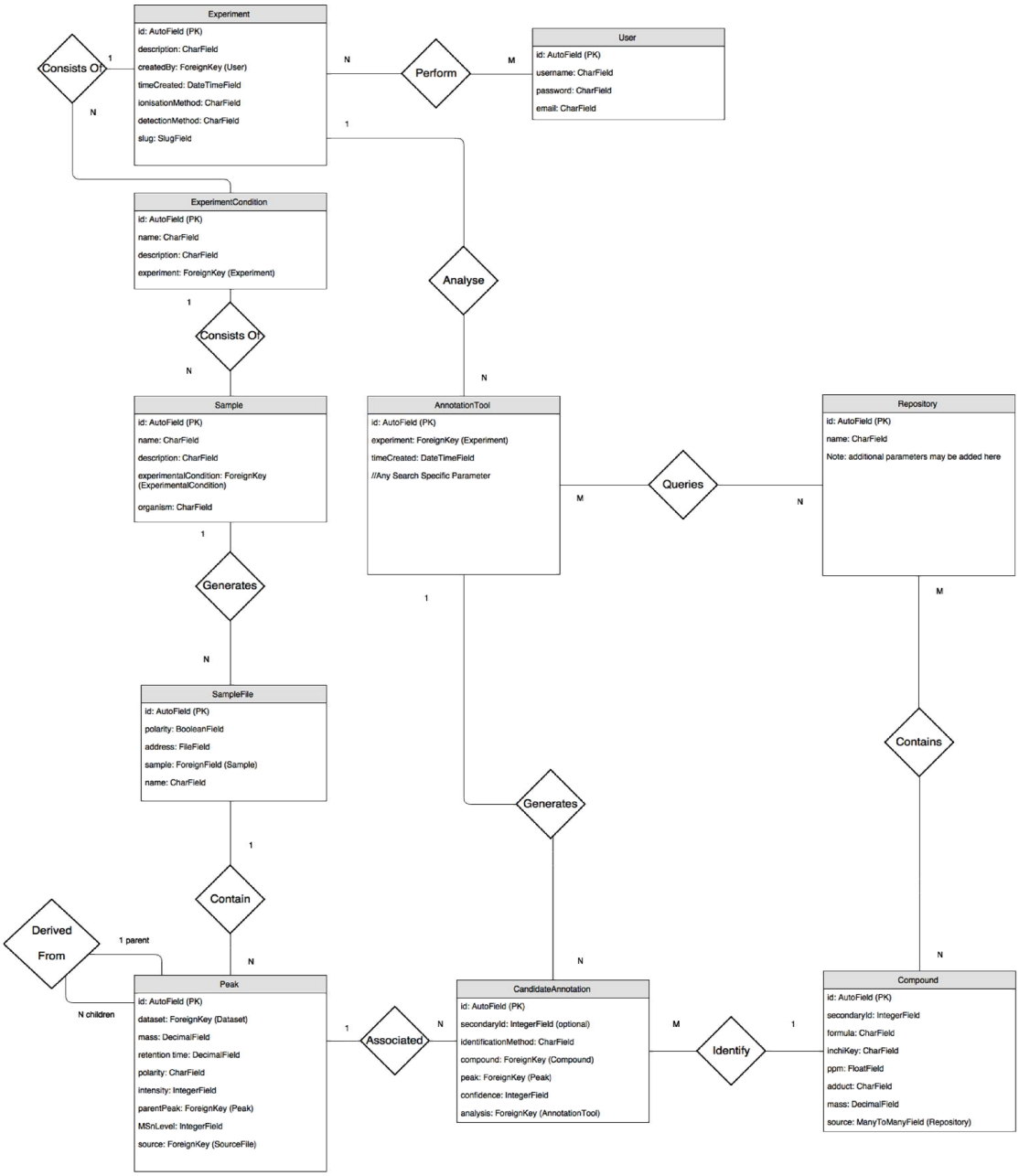
Name:	Specify A Preferred Candidate Annotation
Users:	Scientific Researcher
Goals:	To specify a preferred annotation from the available candidate annotations, thereby proposing the identification of the metabolite corresponding to a peak.

Summary:	The user selects a preferred annotation, adding a justification for selection, which other users may review. The preferred annotation is then associated with the peak in the database.	
Priority:	SHOULD HAVE	
Preconditions:	User has successfully generated or retrieved candidate annotations for a fragmentation set.	
Steps:	User Actions	System Response
	1. Selecting a candidate annotation, the user clicks 'Prefer Annotation'. 3. A justification for the selection of the preferred candidate annotation is entered.	2. Render 'Prefer Annotation' page. 4. Process the form. 5. Associate the preferred candidate annotation to the peak in the database. 6. Render the fragmentation set page displaying the preferred candidate annotation and the associated chemical formula for the associated peak.
Post conditions:	A preferred annotation is associated with the selected peak in the database, alongside the user and the justification for selection.	
Related Use Cases:	Generate Candidate Annotations for Peaks	

Appendix C Design Documentation

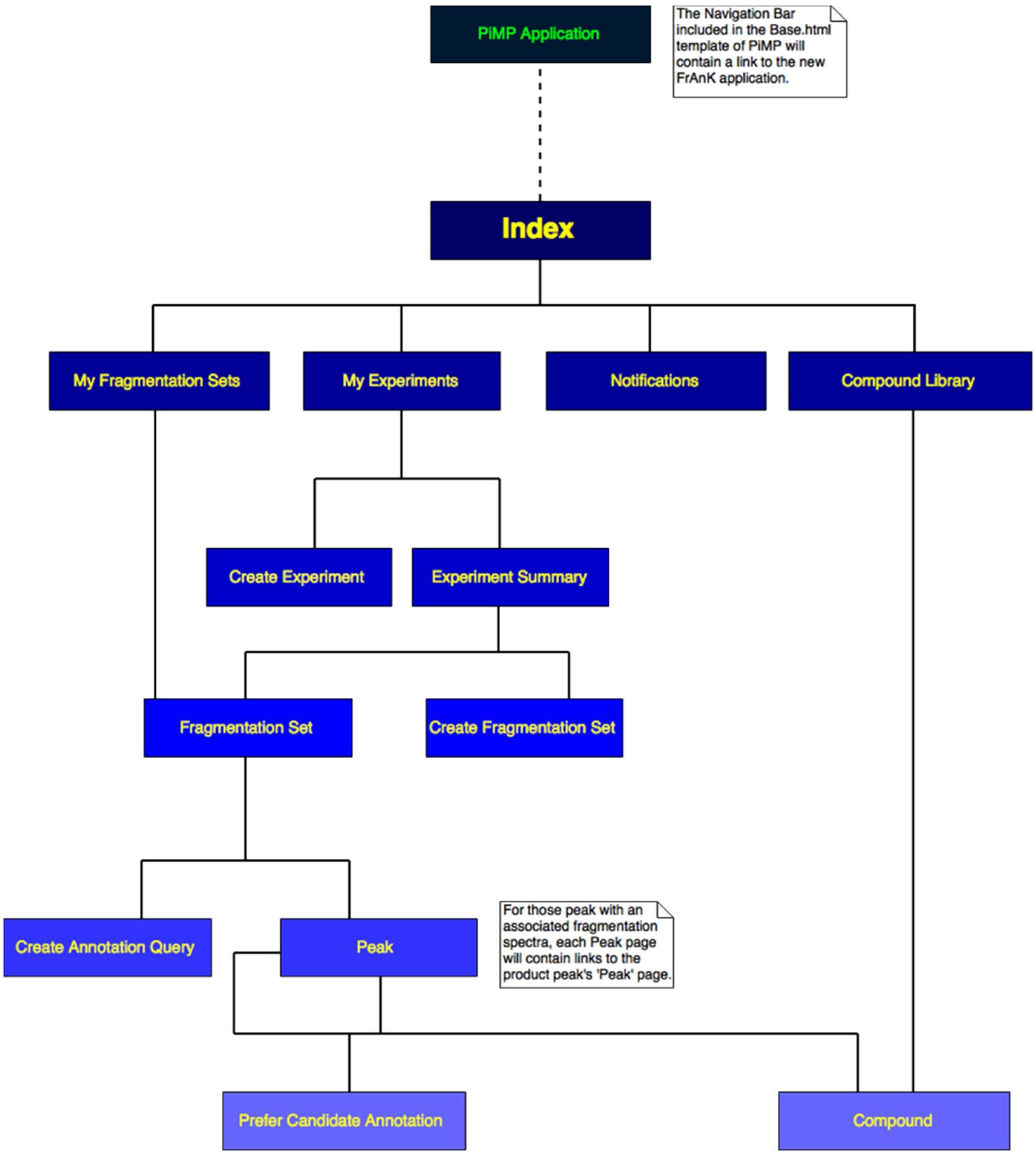
Initial FrAnK Database Schema Derived From ER Model

It should be noted that the Django Framework automatically generates an implicit table for Many-To-Many relationships. Therefore, those tables containing simply two foreign key references have been omitted for clarity.



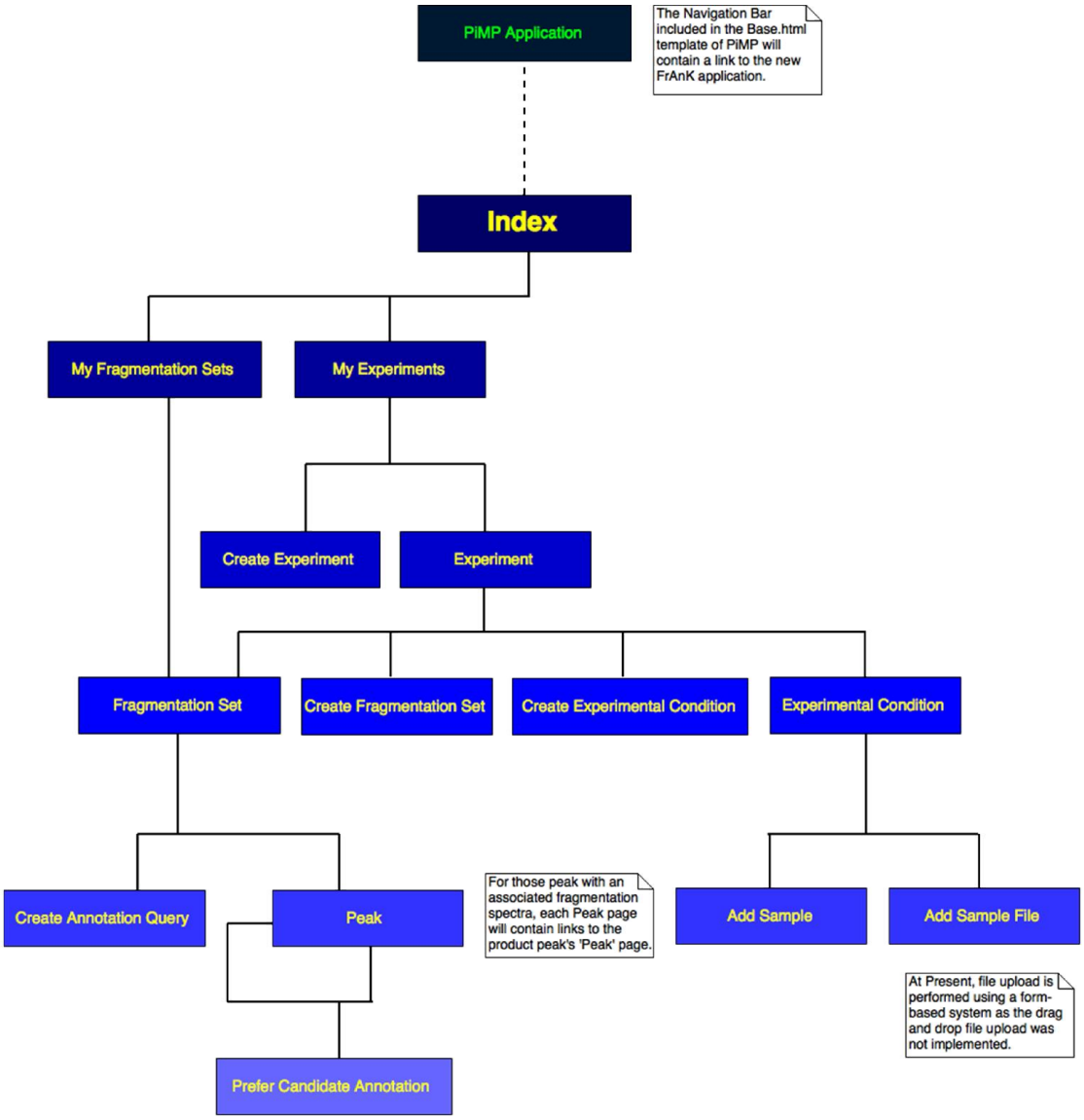
FrAnK Site-Map Design

The following is a site-map which was designed based upon the project requirments and identified use cases.



Current FrAnK Site Map

The following is the site-map corresponding to the current implementation of the application. Due to factors of time, several elements of the original design could not be implemented. These are discussed in additional detail in *section 3.5*.

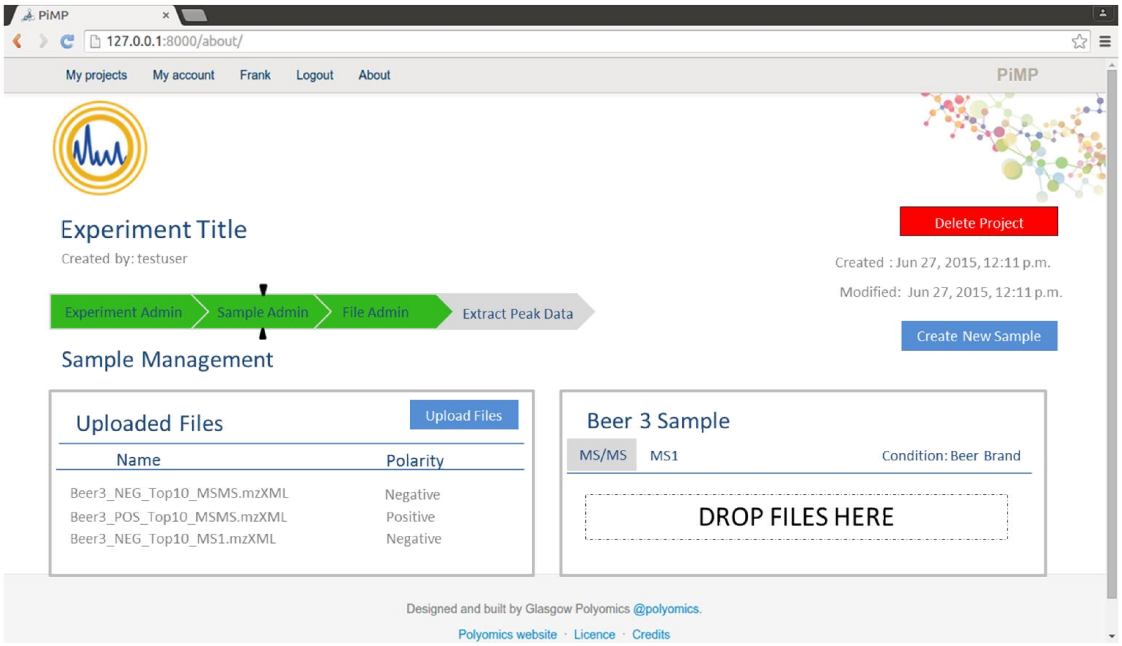


FrAnK URL Mapping

Page/View Name	URL
frank_index	"/frank/"
my_experiments	"/frank/my_experiments/"
add_experiment	"/frank/my_experiments/add_experiment/"
experiment_summary	"/frank/my_experiments/*experiment_slug* /"
add_experimental_condition	"/frank/my_experiments/*experiment_slug* / add_experimental_condition/"
condition_summary	"/frank/my_experiments/*experiment_slug* /*condition_slug* /"
add_sample	"/frank/my_experiments/*experiment_slug* / *condition_slug* /add_sample"
add_sample_file	"/frank/my_experiments/*experiment_slug* / *condition_slug* /*sample_slug* /add_sample_file/
create_fragmentation_set	"/frank/my_experiments/*experiment_slug* / create_fragmentation_set/"
my_fragmentation_sets	"/frank/my_fragmentation_sets/"
fragmentation_set	"/frank/my_fragmentation_sets/*fragmentation_set_slug* /"
define_annotation_query	"/frank/my_fragmentation_sets/*fragmentation_set_slug* / *annotation_tool_slug* /define_annotation_query_parameters/"
peak_summary	"/frank/my_fragmentation_sets/*fragmentation_set_slug* /*peak_slug* /"
make_spectra_plot	"/frank/my_fragmentation_sets/*fragmentation_set_slug* / *peak_slug* /msn_spectra_plot.png/"
specify_preferred_annotation	"/frank/my_fragmentation_sets/*fragmentation_set_slug* /*peak_name* / *annotation_id* /specify_preferred_annotation/"

Wireframe Designs

Experiment Page

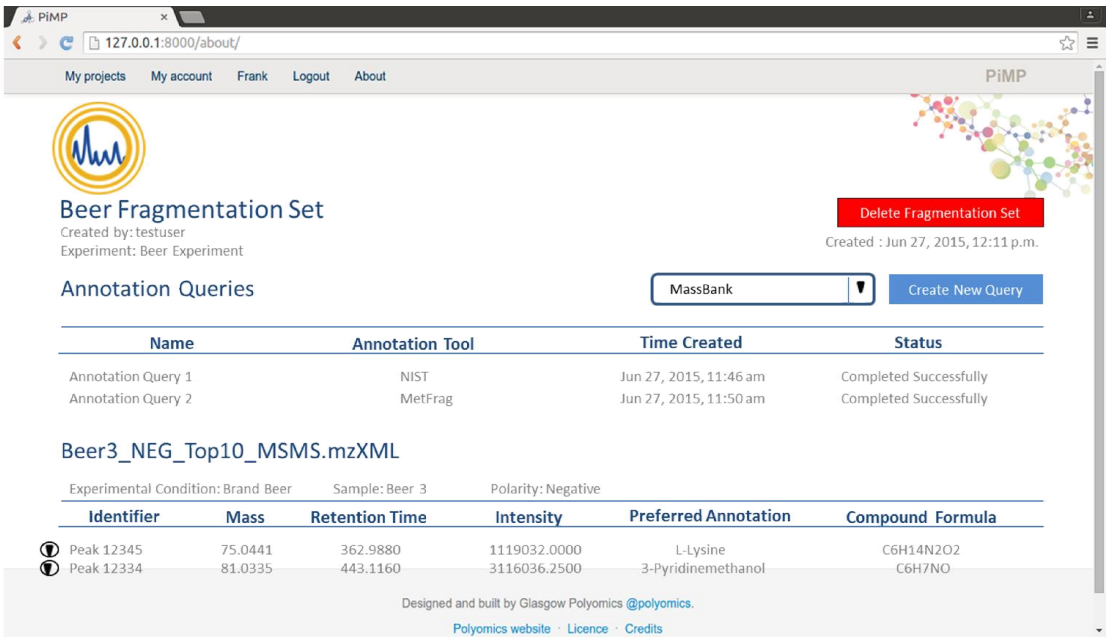


As the PiMP application has been successfully implemented and adopted by the staff of the Metabolomics Facility, it would be logical for the FrAnK application to maintain the already familiar layout to improve user acceptance. As such, the proposed 'Experiment' page of FrAnK follows a similar template to that of the Project set-up of the parent PiMP application. From the 'Experiment Admin' tab, the user may add collaborators to an experiment and add experimental conditions. As such, the initial step in the experimental set-up is the definition of the experimental design. As can be seen from the above wireframe, the 'Sample Admin' tab of the workflow allows the user to create new experimental samples and upload files using the existing PiMP drag-and-drop file upload (accessed from the upload files button in blue). Upon upload, files can be dragged and dropped into the appropriate sample bin indicating whether the file is a fragmentation file (MS/MS) or a distinct full-scan of the MS1 peaks (MS1).

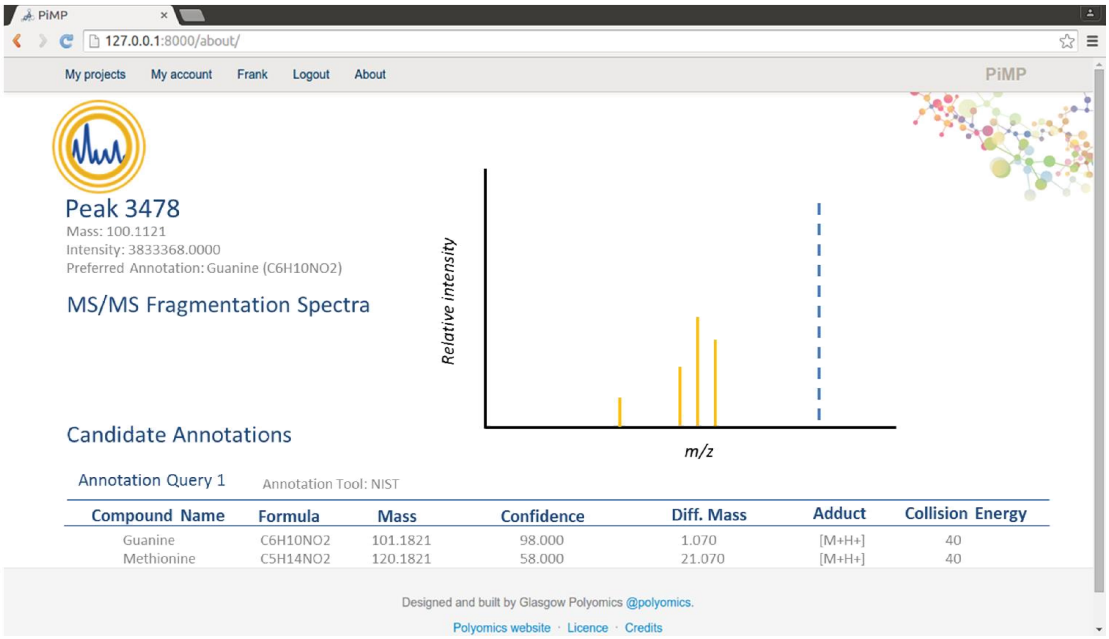
Fragmentation Set Page

The 'Fragmentation Set' page displays the annotation queries, requesting the annotation of the peaks contained in the set, at the top of the page in a tabular manner. In addition, the user can select an annotation tool for the retrieval of candidate annotations via a drop-down menu. The 'Create New Query' button will navigate the user to a page displaying a form specific to the specified annotation tool for declaration of the query parameters. Furthermore, the 'Fragmentation Set' page will display the MS1 peaks contained within the set. The MS1 peaks will be ordered in ascending order by the measured mass to allow for easy lookup by the researcher. For each peak, the mass, retention time and intensity will be displayed. Additional columns will be provided to display the chemical compound and formula associated with a user-specified preferred-annotation for a peak. As the Fragmentation Set may be conceived from

numerous source files, the peaks will be grouped by source file for clarity. The table is intended to be interactive, with the inclusion of a dropdown button corresponding to each peak. Selection of the dropdown button will expand the table to display a sub-table, containing the peak data corresponding to the fragmentation spectra of the peak, indented from the main table. Selection of the drop-down button will collapse the table back to its original state.



Peak Page



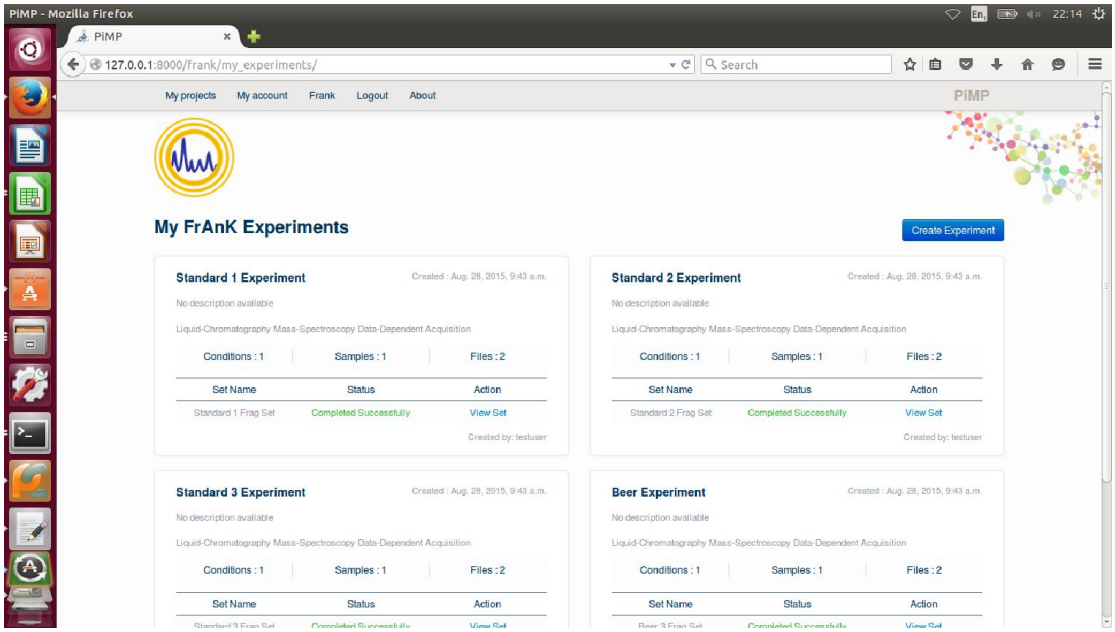
The 'Peak' page displays the relevant MS data for the peak and a graphical representation of the associated fragmentation spectra. In addition to plotting the product peaks of the parent, the graph will plot the precursor peak (dashed blue line of graph) to allow for visual inspection by the researcher. The graph is

intended to serve as an interactive navigation tool, providing links to the 'Peak' pages associated with those peaks comprising the fragmentation spectrum. At the bottom of the page, a table of putative candidate annotations will be displayed, grouped by the annotation query from which they originated. It is intended that the name of the chemical compound identified by each candidate annotation will serve as a link to a 'Compound' page. This would provide additional compound data and in future iterations may display a visual comparison between the observed fragmentation spectrum and that of the spectral reference library from which the candidate annotation was originally retrieved.

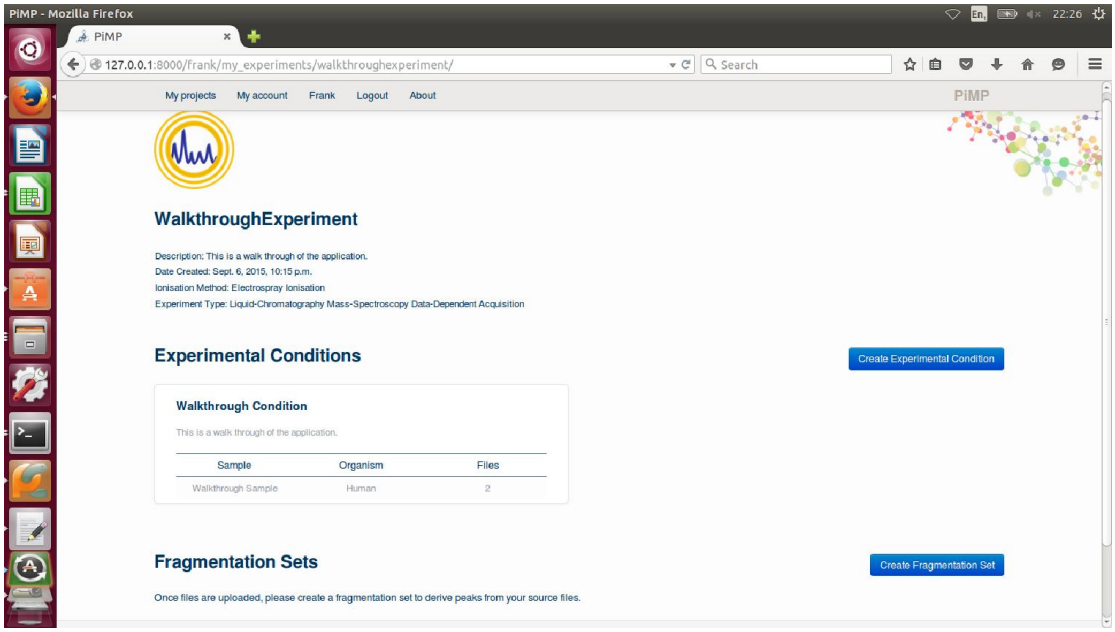
Appendix D Implementation Documentation

Application Screen Dumps

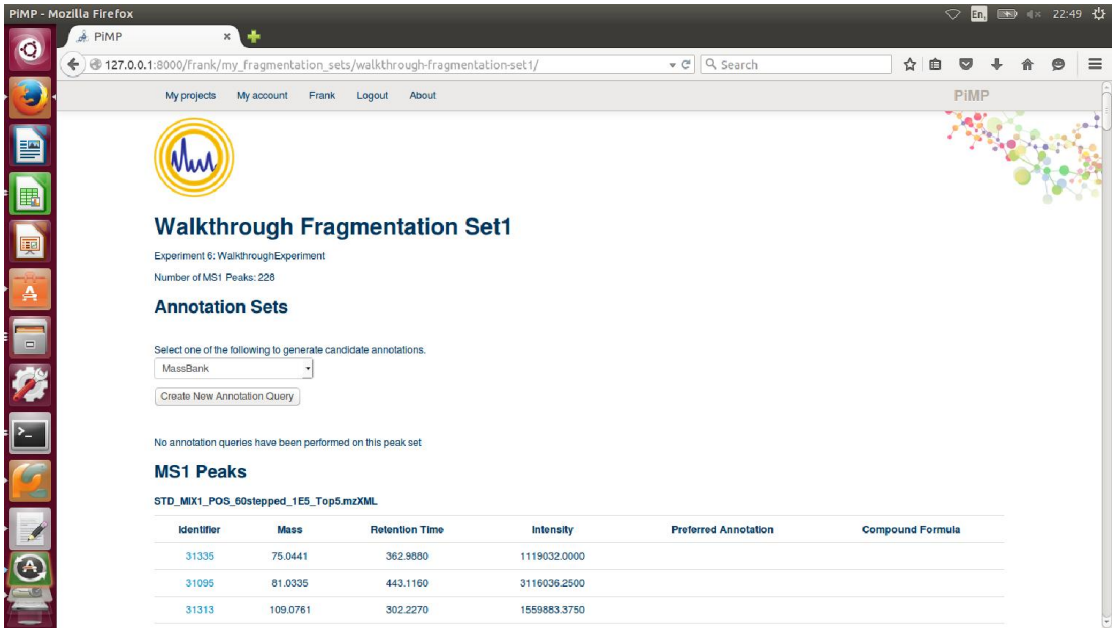
My Experiments Page



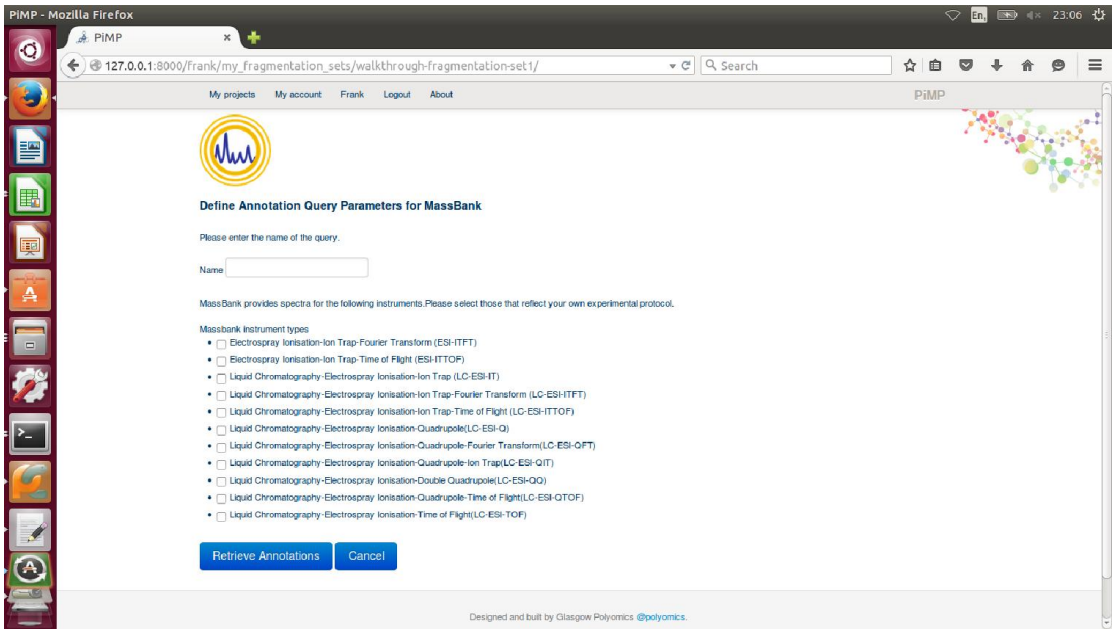
Experiment Page



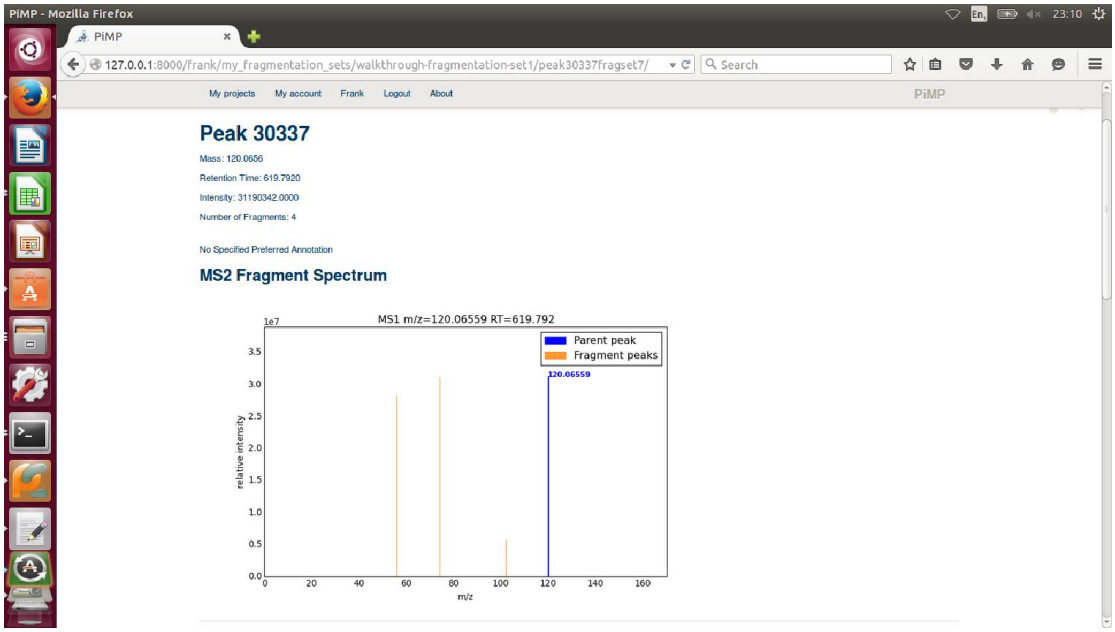
Fragmentation Set Page



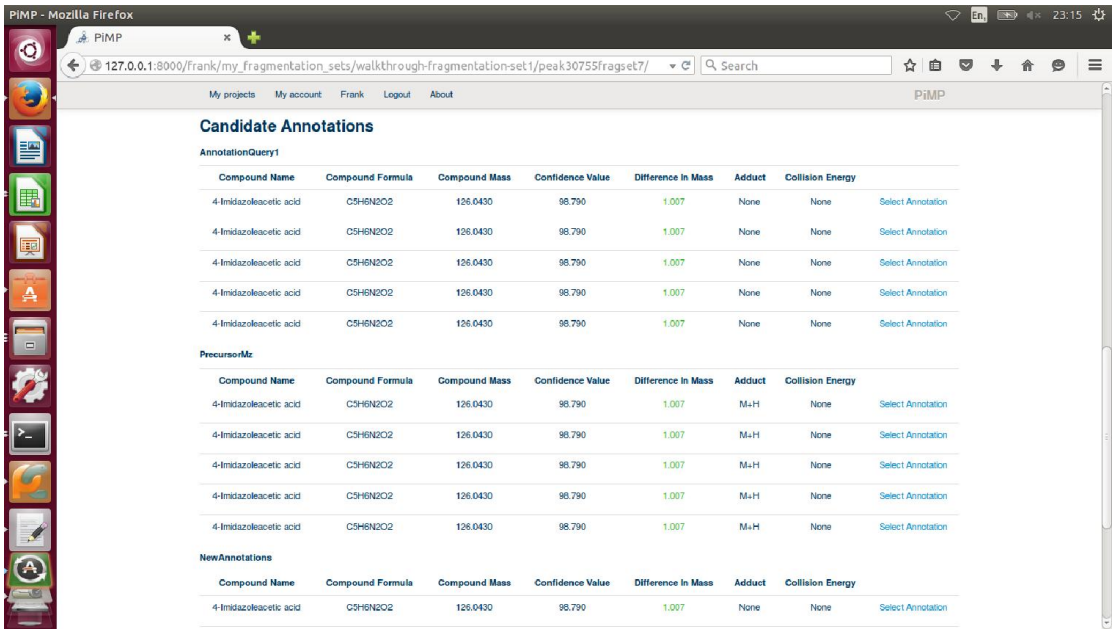
Define Annotation Query Page



Peak Page (Top)



Peak Page (Bottom)



Summary of End-to-End Testing

Name of View	Description of Test	Inputs	Anticipated Outcome	Outcome
index	Ensure page renders for authorised user	Get request	Status code 200	Status code 200
index	Ensure unauthorised user is redirected	Get request	Status code 200	Status code 200
my_experiments	Ensure page renders for authorised user	Get request	Status code 200	Status code 200
add_experiment	Ensure page renders for authorised user	Get request	Status code 200	Status code 200
add_experiment	Test the addition of new experiment	Valid POST request	New experiment added to database	New experiment added to database
add_experiment	Attempt to create experiment with duplicate name	Duplicate POST request	Page renders with form error, experiment not added to database	Page renders with form error, experiment not added to database
experiment_summary	Ensure page renders for authorised user	Get request	Status code 200	Status code 200
add_experimental_condition	Ensure page renders for authorised user	Get request	Status code 200	Status code 200
add_experimental_condition	Test the addition of new experimental condition	Valid POST request	New experimental condition added to database	New experimental condition added to database
add_experimental_condition	Attempt to create experimental condition with duplicate name	Duplicate POST request	Page renders with form error, experimental condition not added to database	Page renders with form error, experimental condition not added to database

condition_summary	Ensure page renders for authorised user	Get request	Status code 200	Status code 200
add_sample	Ensure page renders for authorised user	Get request	Status code 200	Status code 200
add_sample	Test the addition of a new Sample	Valid POST request	New Sample added to database	New Sample added to database
add_sample	Attempt to create a Sample with duplicate name	Duplicate POST request	Page renders with form error, sample not added to database	Page renders with form error, sample not added to database
add_sample_file	Ensure page renders for authorised users	Get request	Status code 200	Status code 200
add_sample_file	Ensure new Sample Files can be uploaded	Valid POST request	Sample file added to database	Sample file added to database
add_sample_file	Attempt to upload a duplicate file to a Sample	Duplicate POST request	Page renders with form error, sample file is not duplicated in database	Page renders with form error, sample file is not duplicated in database
add_sample_file	Attempt to upload an invalid file format	Invalid POST request	Page renders form errors to user	Page renders form errors to user
create_fragmentation_set	Ensure page renders successfully	Get request	Status code 200	Status code 200
create_fragmentation_set	Create a new fragmentation set from an LCMS experiment	Valid Post Request	Peaks extracted and Fragmentation Set added to database	Peaks extracted and Fragmentation Set added to database
create_fragmentation_set	Create a new fragmentation set from an GCMS experiment	Valid Post Request	Peaks extracted and Fragmentation Set added to database	Peaks extracted and Fragmentation Set added to database
create_fragmentation_set	Ensure Fragmentation Set name is unique	Duplicate POST request	Page renders form errors to user	Page renders form errors to user

create_fragmentation_set	Ensure fragmentation set cannot be created without source files	Invalid POST Request	Page renders form errors to user	Page renders form errors to user
fragmentation_set_summary	Ensure page renders successfully	Get request	Status code 200	Status code 200
fragmentation_set	Ensure page renders successfully	Get request	Status code 200	Status code 200
fragmentation_set	Test MassBank can be selected for Annotation Query	Valid Post Request	Status code 200	Status code 200
fragmentation_set	Test NIST can be selected for Annotation Query	Valid Post Request	Status code 200	Status code 200
fragmentation_set	Test Precursor Mass Filter can be selected for Annotation Query	Valid Post Request	Status code 200	Status code 200
peak_summary	Ensure page renders successfully	Get request	Status code 200	Status code 200
define_annotation_query	Ensure page renders successfully - MassBank Selection	Get request	Status code 200	Status code 200
define_annotation_query	Ensure page renders successfully - NIST Selection	Get request	Status code 200	Status code 200
define_annotation_query	Ensure page renders successfully - Precursor Mass Filter Selection	Get request	Status code 200	Status code 200
define_annotation_query	Create valid annotation query (MassBank)	Valid Post Request	Candidate Annotations Retrieved	**TEST FAILED**
define_annotation_query	Create valid annotation query (NIST)	Valid Post Request	Candidate Annotations Retrieved	Candidate Annotations Retrieved

define_annotation_query	Create valid annotation query (Precursor Mass Filter)	Valid Post Request	Candidate Annotations Retrieved	Candidate Annotations Retrieved
specify_preferred_annotation	Ensure page renders successfully	Get request	Status code 200	Status code 200
specify_preferred_annotation	Add a preferred annotation to a peak	Valid Post Request	Preferred annotation added to peak	**TEST FAILED**

Appendix E Evaluation Documentation

Evaluation Tasks

1. Create a new experiment

Step 1 – Select "My Experiments" from the "Home" page.

Step 2 – Select "Add New Experiment" from the "My Experiments" page.

Step 3 – Enter a Title ("Standard 1 Experiment") and Description for the New Experiment. In addition, select the Ionisation Method ("Electron Ionisation Spray") and a Detection Method ("LCMS Data Dependent Acquisition").

Step 4 – Press "Submit" to create the new experiment.

2. Create a new experimental condition

Step 1 – From the "My Experiments" page, select the new experiment ("Standard 1 Experiment").

Step 2 – Select "Add New Experimental Condition"

Step 3 – Enter a Name ("Standard 1 Mixture") and Description for the New Experimental Condition.

Step 4 – Press "Submit" to create the new experimental condition.

3. Create a new sample to the experimental condition

Step 1 – From the "Standard 1 Experiment" page, select the new experimental condition ("Standard 1 Mixture").

Step 2 – Select "Add Sample"

Step 3 – Enter a Name ("Standard 1") and Description for the New Sample. Leave the Organism field blank as the Sample is not biological in nature.

Step 4 – Press "Submit" to create the new sample.

4. Upload new sample files to the sample

Step 1 – From the "Standard 1 Mixture" page, select the "Add Sample File" link beneath the Sample Table for "Standard 1".

Step 2 – Select the polarity of the file ('Positive') and select the file for upload ("Home/ProjectTestData/MS2Data/Standards/STD_MIX1_POS_60stepped_1E5_Top5.mzXML"), then click 'Submit'.

Step 3 – Select the "Add Sample File" link beneath the Sample Table for "Standard 1" for a second time.

Step 4 – Select the polarity of the second file ('Negative') and select the file for upload ("Home/ProjectTestData/MS2Data/Standards/STD_MIX1_NEG_60stepped_1E5_Top5.mzXML"), then click 'Submit'.

5. Create a Fragmentation Set

Step 1 – Return to the "Standard 1 Experiment" via the link at the bottom of the page.

Step 2 – The "Generate New Fragmentation Set" link should now be visible.

Step 3 – Click on the link and enter a name for the Fragmentation Set ("Standard 1 Fragmentation Set") and click submit.

Step 4 – The status of the newly created Fragmentation Set should now be visible.

6. Create a new Annotation Query

Step 1 – Once the status of the Fragmentation Set is "Completed Successfully", the peak data extracted from the source files will now be accessible via a link, please click it.

Step 2 – The "Standard 1 Fragmentation Set" page should show, via a table, the MS1 peak data for the source files (note, this is not a complete listing, rather those MS1 peaks with fragmentation data associated with them).

Step 3 – From the drop-down menu at the top, please select an Annotation Tool to use to annotate the fragmentation spectra ('NIST'). Then click "Create New Annotation Set".

Step 4 – From the page please enter the parameters for the search….

Name: *Standard 1 Annotation Query*

Maximum Number of Hits: *10*

Search Type: *Generic Search MS/MS Search in MS/MS Library*

Libraries: *Select "Tandem (MS/MS) Library – Small Molecules"*

Select "Tandem (MS/MS) Library – Biologically Active Peptides"

Step 5 – Click "Submit", the status of the new Annotation Query should be displayed.

Step 6 – Once the status of the new Annotation Query is "Completed Successfully", the candidate annotations can be viewed by click on a Peak link.

7. Specify a Preferred Annotation

Step 1 – From the "Standard 1 Fragmentation Set" page, select one of the peak identifiers (which serve as links).

Step 2 – At the top of the "Peak" page is a graphical representation of the fragmentation spectra associated with the peak. Beneath the graph of the fragmentation spectra, is a table of the peaks which comprise the fragmentation spectra, which can be used to link to different levels of MSn data. At the bottom of the page should be a table of candidate annotations returned by the NIST Annotation Query. To "prefer" a candidate annotation, click on the "Select Annotation" link in the right-most column of the table.

Step 3 – The "Preferred Annotation" page should render. At this juncture, a justification for the selection of the annotation can be provided. Click "Submit".

Step 4 – From the "Peak" page, return to the "Standard 1 Fragmentation Set" page using the link at the bottom of the page.

Step 5 – In the table of MS1 peaks associated with the Fragmentation Set, should now appear the name of the chemical associated with the preferred annotation and its chemical formula.

Interview Questions - Client

1. What are your general impressions of the development of the application to date?

2. Do you think the functionality of the application would be valuable to the research staff at the facility?

3. How would you rate the overall usability of the application's user interface?

4. Do you think the text descriptions of the various links and fields accurately convey their purpose? Were there any descriptions in particular that you found to be ambiguous or confusing?

5. Does the application present the researcher with all the necessary data required to evaluate the candidate annotations and fragmentation spectra? Is this conveyed in an appropriate manner?

6. Do you have any recommendations or suggestions for improving the user interface?

7. Were there any points during the evaluation tasks that you became stuck? If so, could you could you elaborate upon these?

8. Has the development of the application achieved what you envisaged to be the outcomes of the project?

9. Are there any key elements of functionality you would like to have been added to the application over the course of the project?

10. Were you satisfied with the process used to develop the application? In particular, was the prototyping of the application valuable to you?

11. Do you believe that the current application provides the flexibility required for future development?

12. What future development of the application do you envisage?

Qualitative Evaluation of Candidate Annotations

	Reference Compound			Candidate Annotation			
Compound	Name	Formula	Mass	Measured Mass	Name	Formula	Confidence
StdMix1_11	Inosine	C10H12N4O5	268.0807695	267.0738	Ionsine	C10H12N4O5	92.58
StdMix1_14	L-Tryptophan	C11H12N2O2	204.0898776	205.0970	L-Tryptophan	C11H12N2O2	98.99
StdMix1_15	2-Phenylglycine	C8H9NO2	151.0633285	152.0706	DL-alpha-Phenylglycine	C8H9NO2	98.89
StdMix1_17	L-Methionine	C5H11NO2S	149.0510493	150.0583	L-Methionine	C5H11NO2S	97.17
StdMix1_18	Guanine	C5H5N5O	151.0494098	152.0567	Guanine	C5H5N5O	98.98
StdMix1_2	Imidazole-4-acetate	C5H6N2O2	126.0429274	127.0502	4-Imidazoleacetic acid	C5H6N2O2	98.79
StdMix1_27	N2-Acetyl-L-lysine	C8H16N2O3	188.1160924	187.1089	N.alpha-Acetyl-L-lycine	C8H16N2O3	50.04
StdMix1_3	N-Acetyl-D-glucosamine	C8H15NO6	221.0899372	222.0971	N-Acetyl-D-glucosamine	C8H15NO6	95.76
StdMix1_35	L-Citrulline	C6H13N3O3	175.0956913	176.1030	L-Citrulline	C6H13N3O3	98.72
StdMix1_41	D-Glucosamine	C6H13NO5	179.0793725	179.0790	Galactosamine	C6H13NO5	84.63
StdMix1_43	L-Histidine	C6H9N3O2	155.0694765	156.0768	L-Histidine	C6H9N3O2	98.69
StdMix1_46	L-Arginine	C6H14N4O2	174.1116757	175.1189	DL-Arginine	C6H14N4O2	49.10
StdMix1_5	Melatonin	C13H16N2O2	232.1211778	233.1285	Aminoglutethimide	C13H16N2O2	36.65
StdMix1_50	4-Coumarate	C9H8O3	164.0473441	163.0402	trans-2-Hydroxycinnamic acid	C9H8O3	36.59
StdMix1_52	Phthalate	C8H6O4	166.0266087	167.0339	1,2-Benzenedicarboxylic acid	C8H6O4	97.49
StdMix1_54	Methylmalonate	C4H6O4	118.0266087	117.0546	Succinic acid	C4H6O4	98.45
StdMix1_68	Orotate	C5H4N2O4	156.0171066	155.0099	Orotic Acid	C5H4N2O4	21.78
StdMix1_69	MOPS	C7H15NO4S	209.0721787	210.0795	4-Morpholinopropanesulfonic acid	C7H15NO4S	97.08
StdMix1_72	D-Galacturonate	C6H10O7	194.0426527	193.0355	D-Glucuronic acid	C6H10O7	98.95
StdMix1_98	3-Hydroxyphenylacetate	C8H8O3	152.0473441	152.1070	Methylparaben	C8H8O3	83.83
StdMix2_1	Hypoxanthine	C5H4N4O	136.0385108	137.0458	Hypoxanthine	C5H4N4O	98.99
StdMix2_10	Nicotinamide	C6H6N2O	122.0480128	123.0553	Niacinamide	C6H6N2O	95.09
StdMix2_18	6-Methylaminopurine	C6H7N5	149.0701453	150.0774	Adenin, N6-methyl	C6H7N5	97.61
StdMix2_20	Pyridoxal	C8H9NO3	167.0582432	168.0655	Pyridoxal	C8H9NO3	98.98
StdMix2_25	L-2-Aminoadipate	C6H11NO4	161.0688078	162.0759	Hexanedioic acid, 2-amino-	C6H11NO4	93.94
StdMix2_36	cytosine	C4H5N3O	111.0432618	112.0504	Cytosine	C4H5N3O	97.14
StdMix2_37	N-Acetylornithine	C7H14N2O3	174.1004423	175.1075	N-.alpha.-Acetyl-L-ornithine	C7H14N2O3	97.91
StdMix2_44	Choline phosphate	C5H14NO4P	183.0660445	184.0731	Phosphocholine	C5H14NO4P	96.82
StdMix2_51	L-Ornithine	C5H12N2O2	132.0898776	133.0972	DL-Ornithine	C5H12N2O2	53.77
StdMix2_61	Mesaconate	C5H6O4	130.0266087	131.0703	Butanedioic acid, methylene-	C5H6O4	98.90

Appendix F README

The following are guidelines for the running of the FrAnK application...

FrAnK was developed on an HP 15 Laptop, consisting of an Intel Core i3-45005U processor and 7.7 GB RAM, running the Ubuntu 15.04 operating system. PiMP, including the dependencies described in section 2.4.2, was installed. To provide clarity for future developers, additional packages were installed such as mysql-server 5.6.25, r-base-core 3.1.2-2, wine 1.6.1:1.6.2, rabbitmq-server 3.2.4-1 and oracle-java8 in addition to those stated in the 'requirements_frank.txt' document of the application. The querying of MassBank is performed through Suds, a python package providing a client for the sending and retrieval of SOAP requests. The package should be included in the 'requirements_frank.txt' file.

In order to run the application, the set-up guide included within the PiMP application should be followed, including the installation of any dependencies such as 'mzmatch.R'. However at this juncture, the population script of the application should not be run. Whilst using the virtual environment created in the PiMP setup guide, ensure that the command 'pip install –r requirements_frank.txt' is also run to install the dependencies for the new application.

To perform analysis in PiMP and the retrieval of candidate annotations from FrAnK, it is highly recommended to future developers that mysqlserver and rabbitmq are installed to allow for asynchronous processing as the SQLite database commonly used in Django application development does not support concurrent transactions. In addition to MySQL and celery, wine should be installed. Upon installation, the mapping of the wine software to the file hierarchy of the local system by the command line argument 'winecfg' and the selection of the appropriate tab at the top of the GUI display. Having gained familiarity with which directory corresponds to the 'C:\\' in wine, NIST 14 can be installed to this location. In order to query the spectral reference libraries of NIST 14, you will require the MS PepSearch program which must also be installed through wine. The 64-bit version of MS PepSearch for Windows should be compatible with the wine software.

It is recommended that once the installation of the dependencies associated with FrAnK have been installed that the population script, 'populate_pimp.py' be run to provide default values for both the PiMP and FrAnK applications. However, prior to running the script, the hardcoded paths required as defaults for the AnnotationTools should be amended to reflect the local installation. As such, were a local installation of MassBank to become available, the address passed to the SOAP client could easily be altered to a local port.