TALLINN UNIVERSITY OF TECHNOLOGY

School of Information Technologies

Department of Software Science

Jaak Kütt 200814IAIB

Georg Margus 193294IAIB

Lauri Kask 193685IAIB

SOFTWARE FOR PROTEIN DETERMINATION

Bachelor Thesis

Supervisor

Priit Järv

PhD

Tallinn 2022

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Jaak Kütt 200814IAIB
Georg Margus 193294IAIB
Lauri Kask 193685IAIB

Tarkvara Valkude Määramiseks

Bakalaureusetöö

Juhendaja
Priit Järv
PhD

Tallinn 2022

Author’s Declaration of Originality

We hereby certify that we are the sole authors of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author:	Jaak Kütt, Georg Margus, Lauri Kask	....
		(signature)
Date:	May 30th, 2022	

Annotatsioon

Bakalaurusetöö raames loodi Tallinna Tehnikaülikooli Küberneetikaosakonna Süsteemibioloogia laborile töölauarakendus. Rakenduse eesmärk on senise geelianalüüsi tööprotsessi optimiseerimine, kiirendades teatud toiminguid mis nõudsid ebavajalikul määral korduvaid liigutusi ja täpsust.

Tarkvara ehitati Qt raamistikule. Rakendus loodi Pythonis, kasutades PySide6 mähisteeki. Andmetalletuseks on valida SQLite ja PostgreSQLi andmebaaside vahel, ning piltide laadimiseks failisüsteemi või OMERO andmebaasi.

Tulemuseks on tarkvaralahendus, mis suudab geelipilte sisse laadida, pildilt alamosa valida ning seda pöörata, ning teostada taustaeemaldust. Pildile saab paigutada radu, mille põhjal koostatakse intensiivsusgraafe, mida saab vajadusepõhiselt piiritleda nõutud pindala kättesaamiseks. Tulemusena saadud andmeid saab talletada andmebaasi hilisemaks ülevaatluseks ja analüüsiks.

Alguses planeeriti valminud rakendus avaldada publikule avatud lähtekoodiga, aga lõpupoole selgus, et rakendus pole veel piisavalt valmis, nõudes lisaviimistlusi.

Projekti GitLabi salv on leitav aadressilt `https://gitlab.cs.ttu.ee/jakutt/iaib`.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 30 leheküljel, 8 peatükki, 25 joonist, 9 tabelit.

Abstract

As part of this thesis, a desktop application for Tallinn University of Technology's Cybernetics Institute's Systems Biology laboratory was made. The goal was to optimize the current protein analysis workflow, speeding up certain tasks and operations that required unnecessary repetition and precision.

The software was built on the Qt framework. It was written in Python, using the PySide6 wrapper library. It uses either SQLite or PostgreSQL for data storage and the filesystem or OMERO for images.

The result is a software that can load gel images and crop, rotate, perform background subtraction on them. Lane regions can be defined on the image for intensity plot generation, which can be cut off at places to extract desired areas. Those results can be stored in a database for later review and analysis.

The software was planned to be released to the public as an open-source project. Towards the end of development, however, it was considered not to be ready yet, requiring more polish.

The project's GitLab repository can be found at `https://gitlab.cs.ttu.ee/jakutt/iaib`.

The thesis is in English and contains 30 pages of text, 8 chapters, 25 figures, 9 tables.

List of Abbreviations and Terms

API	Application Programming Interface
COX4	Cytochrome c oxidase subunit 4, mitochondrial
CRUD	Create, Read, Update, Delete
CSS	Cascading Style Sheets
CSV	Comma-separated values
DBMS	Database Management System
GPLv3	GNU General Public License version 3
GUI	Graphical User Interface
HTML	Hypertext Markup Language
ORM	Object-relational mapper
OS	Operating system
ROI	Region of Interest
SQL	Structured Query Language
SSL	Secure Sockets Layer
UI	User interface
UX	User experience

Table of Contents

List of Figures viii

List of Tables ix

1 Introduction 1

1.1 Domain Overview . . . 2

1.1.1 Protein Determination . . . 2

1.1.2 Gel . . . 2

1.1.3 Gel Processing . . . 2

1.1.4 Gel Analysis . . . 3

1.1.5 Data Analysis . . . 3

1.2 Project Scope . . . 3

2 Project Description 4

2.1 Problem Outline . . . 4

2.2 Requirements Analysis . . . 4

2.2.1 Initial Outline . . . 5

2.2.2 Use Case Analysis . . . 5

2.2.3 User Flow Validation . . . 6

2.2.4 Identified User Stories . . . 7

2.3 Existing Solutions . . . 7

2.3.1 ImageJ . . . 7

2.3.2 Image Lab . . . 8

2.4 Project Management . . . 8

3 Project Design 9

3.1 Frameworks and Libraries . . . 9

3.2 Architectural Choices . . . 11

3.2.1 Dependency Injection . . . 11

3.2.2 Events . . . 12

3.2.3 Undo/redo . . . 12

3.2.4 Database . . . 13

3.3 Measuring User Confidence . . . 14

4 Software Description 15

4.1 Prerequisites and Installation . . . 15

4.1.1 Application Startup 16
4.2 Application Layout 17
4.2.1 Toolbar 17
4.2.2 Navigation 17
4.2.3 Content Area 17
4.3 Application Views 17
4.3.1 Gel List View 18
4.3.2 Gel Detail View 18
4.3.3 Measurement Types View 18
4.3.4 Gel Image Raw View 19
4.3.5 Gel Image Adjust View 19
4.3.6 Gel Image Background View 19
4.3.7 Gel Image Lanes View 20
4.3.8 Gel Image Measurements View 20
4.3.9 Settings View 21

5 Validation **23**

6 Results **24**
6.1 Internationalization 24
6.2 CSV Export 25
6.3 Curved Lanes 25
6.4 Gel Filtering 25

7 Comments and Discussion **26**
7.1 Domain Knowledge 26
7.2 Project Management 26
7.3 Curved Lanes 27
7.4 Qt 28
7.5 Python 28
7.6 Application Architecture 28
7.7 Alternative Image Sources 29

8 Conclusion and Future Development **30**

Bibliography **31**

Appendix 1 – UI Draft **34**

Appendix 2 – User Stories **38**

Appendix 3 – Application Views **40**

Appendix 4 – Requirements Comparison **46**

Appendix 5 – Questionnaire Results **48**

Appendix 6 – Database Schema **50**

Appendix 7 – Non-exclusive licence for reproduction and publication of a graduation thesis **51**

List of Figures

1 *Change in the order of associations between components* 7

2 *Software design.* 9

3 *Cropping/rotation selection ROI.* 19
4 Gel image graph and lanes plot 20
5 *Integration limit lines* 21

6 *Application layout.* 34
7 *Application gels list view.* 34
8 *Application gels single view.* 35
9 *Application raw image view.* 35
10 *Application adjust image view.* 36
11 *Application gel background view.* 36
12 *Application gel lanes view.* 37
13 *Application gel measurements view.* 37
14 *Application gel list view.* 40
15 *Application gel lanes popup view.* 41
16 *Application measurement types list view.* 41
17 *Application gel detail view.* 42
18 *Application image selection popup view (local folder).* 42
19 *Application image raw view.* 43
20 *Application image adjust view.* 43
21 *Application image background subtraction view.* 44
22 *Application image lanes view.* 44
23 *Application image measurements view.* 45
24 *Application settings view.* 45
25 *Application database entity relationship diagram.* 50

List of Tables

1 Time costs of gel analysis 4

2 Steps to add alternative database support 10

3 Requirements of the Systems Biology lab project lead 38
4 Requirements of scientist working at the Systems Biology lab 38
5 Requirements of scientist not working at the Systems Biology lab 38
6 Requirements of a scientist 39
7 Requirements satisfied by alternative applications 46
8 Questionnaire: user roles 48
9 Questionnaire: user confidence 48

1. Introduction

The main motivation for this project was the client's need to analyze protein concentrations efficiently and conveniently. Clients understood the problem well since this is a rather common task in molecular biology. Existing solutions had certain limitations which made them either inefficient or lacking in necessary functionality. We tried to provide a solution that would overcome these limitations.

Protein quantification is a method for finding protein concentrations from blots. The Systems Biology lab currently uses ImageJ for this purpose. The blot image can be adjusted by cropping it, adjusting brightness, and performing background subtraction. Lanes are identified on the processed image and marked with selection tools and defined as lanes. The lanes can then be plotted as histograms, where peaks or valleys indicate band intensities. Those indentations can be closed off with lines, then the enclosed areas selected and exported [1].

There were a few problems with using ImageJ for this:

- accidentally creating a wrong lane selection is not undoable, meaning reopening the project from the last saved state and doing everything again up to that point is necessary;
- the data needs to be manually exported and inserted into a database;
- there can only be a single item in the undo/redo buffer, meaning more than one mistake, in general, requires restarting.

Due to the problems with using ImageJ, the Systems Biology lab requested an application to be made specifically for protein quantification that would resolve them, by providing better undo/redo functionality, a direct database connection, and ability to edit selections and recalculate changes. The application was written in the Python programming language using the PySide6 framework.

1.1 Domain Overview

1.1.1 Protein Determination

Protein determination is the process of finding out the concentration of a protein in a sample [2].

1.1.2 Gel

A gel is a quasi-solid with a wide range of possible properties. Its cross-linked system makes it weigh like a liquid, act like a solid and contributes to its hardness and adhesiveness [3]. There are various types of gels, of which typically relevant to gel electrophoresis are agarose, polyacrylamide and starch. Gel electrophoresis for proteins commonly uses polyacrylamide gels [4].

In gel electrophoresis, gels have wells for the samples, each well run forming a lane. Some lanes, usually one, may be used as a reference, utilizing a commercially available matter for comparison with samples. As voltage is applied, the samples migrate through the lane, forming separate bands within it [4].

1.1.3 Gel Processing

For protein detection, western blotting is a widely used analytical technique involving gel electrophoresis. Proteins in the sample are first separated through gel electrophoresis, then transferred from the gel onto a membrane, most commonly using the electroblotting method. The protein on the membrane is then stained so that it can be visualized, which is used to check the uniformity of the transfer and perform normalization. There are many methods for staining, with Coomassie brilliant blue R 250 and Ponceau S, among others, commonly used. In order to prevent interactions between the membrane and antibody, blocking is done with a protein solution. The membrane is then probed for the protein of interest in a process called incubation, which, when successful, should produce an indicating color. In the detection and visualization phase, the failed probes will be washed away, and a variety of detection methods can be applied for protein level determination. The banding patterns are usually documented by photographing or scanning the gel, which can then be used for further analysis [4]. The image should be taken such that the bands look sharp, without indistinct edges or fuzziness [5].

1.1.4 Gel Analysis

Band quantification is the process of measuring signal intensity of protein bands. Because the intensity is proportional to target protein concentration, it is possible to compare samples with other samples or the control. This data can then be used for further statistical analysis [5].

Measuring by Blot Image

The grayscale blot image can be loaded into an image processing software for analysis. Region of Interest (ROI) can be defined on the image around the lanes/bands subject to analysis. The pixel values in the region of interest are added up in top-to-bottom chunks, with the values from the chunks forming an intensity graph. The difference in areas between the graph and a defined baseline determines the intensity of the given region, which can then be compared with other regions.

1.1.5 Data Analysis

Results from western blotting are commonly used in biochemistry to study different aspects of proteins and disease diagnosis [6]. It can be used, for example, to detect tissue factor in animals [7], act as a confirmatory test for syphilis [8], and for studying asthma [9].

1.2 Project Scope

The given project aims to help with processing the results of western blots through image manipulation and image data analysis, as well as automatically storing analysis results in a database. Source images can be cropped, rotated, have their background subtracted, and color-mapped for ease of use. Sections of the image can have lane regions defined, which will produce intensity graphs based on pixel values within the lane region. The intensity graphs can have hand-applied zero-lines to calculate the desired area. The created lanes, zero-lines, and lane measurements are saved to a database in a form that allows for further analysis with additional tools and retroactive changes.

2. Project Description

The goal was to develop an application that could automate the analysis of gels based on their protein content. The application was supposed to overcome certain limitations of alternative solutions. The clients also imposed certain requirements for the used technology stack and the type of application.

2.1 Problem Outline

The primary issue with the current method are its inefficiencies: the tools used to perform image analysis and store the results have a lot of repetitive, manual steps and require precision and consistency to avoid losing significant chunks of progress. Table 1 outlines the current steps and the reported time taken to perform them for one experiment, keeping in mind that making a mistake requires previous steps to be redone. Since at least three experiments are done, the total time taken can exceed 7.5 hours per protein.

Order	**Action**	**Time taken**
1	Signal analysis	10 minutes
2	Ponceau analysis (normalization)	10 minutes
3	Cytochrome c oxidase subunit 4, mitochondrial (COX4) analysis (normalization by another protein)	10 minutes
4	Upload data to database	120 minutes
		150 minutes

Table 1. Time costs of gel analysis

Ponceau, signal, and COX4 analyses were performed using ImageJ. Data upload was done through a web interface for textual data and OMERO for images. The main pain points were using ImageJ for the analysis steps due to quirks of the software and data insertion through the web interface for the database, which was slow primarily from having to pick out relevant values from ImageJ results and performing repetitive creation/insertion operations using the web interface.

2.2 Requirements Analysis

The requirements for the project were gathered in three phases: 1) initial outline and domain knowledge transfer, 2) use case analysis, 3) user flow validation through visual design.

2.2.1 Initial Outline

Initial meetings with the clients introduced the general technical and non-technical requirements for the software. Two significant goals were set out for the project. Primarily, the software should improve the speed of analysis done at the Systems Biology lab. Secondarily it should be published as open-source software under the GNU General Public License version 3 (GPLv3) to increase adoption by the scientific community.

To that end the first major requirement was that the software should be a desktop application to ease distribution and minimize the overhead required for individual scientists or labs using it. Since the clients wanted to retain the ability to maintain the software tools that they use, it was recommended that the project would be based on the Qt framework[1] which they were familiar with. It was agreed that since the authors were not that familiar with C++ which is native to Qt, a Python middleware framework would be used.

The software was supposed to support individual analysis work and a shared setting - the latter was modeled after the need to integrate with the client's existing systems: 1) PostgreSQL database, 2) OMERO image repository[2].

This divide helped to identify different interested parties from whose perspectives were used to derive the requirements for the project: 1) project lead at the Systems Biology lab, 2) scientist working at the lab, 3) scientist working outside of the lab.

The only existing alternative software found during this phase was ImageJ, which was subsequently tested, and its documentation was used as a means to help derive a common domain vocabulary with the client.

2.2.2 Use Case Analysis

Three main use cases were identified: 1) preparing for analysis, 2) analyzing gel images, and 3) reviewing prior analysis results. Since it was concluded that the scientist working with the software would want to switch between these activities freely, a single user role representing the scientist was enough to describe the use cases.

[1]https://www.qt.io/product/framework
[2]https://www.openmicroscopy.org/omero/

Preparing for Analysis

The user should be able to describe the general context around the analysis, implemented as data table views. This includes:

- describing the physical gel object and the samples added to its lanes;
- describing which types of measurements were done on which gel images.

Analyzing Gel Images

This is the primary use case of the software - the results obtained here are used in further scientific research outside of the application.

The user should be able to select and modify the gel image on which to perform the analysis. Modifications include: 1) crop and rotate the area of interest, 2) subtract image background, and 3) specify the visual false-color mapping for the image. The user should be able to indicate where on the image specific lanes are positioned and how they are shaped. The user should be able to indicate the areas on which the pixel intensity analysis is performed, what is considered the signal, and what the background is, compare, and annotate the analysis results.

Reviewing Prior Analysis

The user should be able to view all previously defined general data and image analysis results without the fear of accidentally modifying anything. It was agreed that a button to toggle between the *viewing* and *editing* modes should always be present on the toolbar, clearly indicating which mode is currently active. The user should not be able to perform any actions which may result in data being modified when not in the editing mode. To simplify the review process the user should be able to search for the previously analyzed gels and measurements.

2.2.3 User Flow Validation

A month after the start of the project a longer meeting with the clients was held to finalize the use case details. As a result a mock-up of the application user interface (UI) views was drawn and finalized over several iterations of feedback from the clients. The visual look and user experience (UX) elements described with it also became part of the requirements. The mock-up can be seen in Appendix 1.

As a result of working through a visual means, a previously missing case of the main analysis flow was discovered leading to substantial changes in the existing implementation.

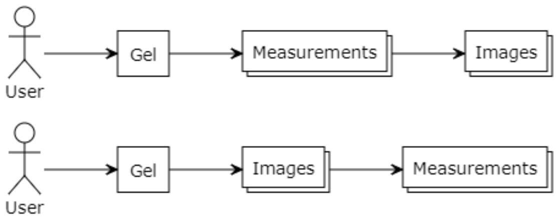


Figure 1. *Change in the order of associations between components*
Top: initial order; Bottom: final order

2.2.4 Identified User Stories

Together with the initially identified interested parties and the requirements found for the scientist role during use case analysis, a total of 39 requirements were identified and described as user stories in tables 3 - 6 of Appendix 2. The stories were divided as:

- three from the perspective of the Systems Biology lab project lead;
- three from the perspective of a scientist working at the Systems Biology lab;
- four from the perspective of a scientist not working at the Systems Biology lab;
- 24 from the perspective of a scientist.

2.3 Existing Solutions

The main alternatives to our application are ImageJ and Image Lab. Neither of them fully covers the client's needs. A comparative table based on the requirements specified in the previous section can be found in Table 7 of Appendix 4.

2.3.1 ImageJ

The existing (partial) alternative to our solution is ImageJ which is an image processing and analysis software for scientific usage. It lacks certain functionalities that our application aims to provide. Firstly, it does not offer the possibility to undo any actions during analysis. Mistakes made at later stages of analysis mean starting the process over again. Several hours of wasted time are possible because of that limitation. Our application does allow the user to correct their errors without discarding the whole analysis. Also, ImageJ does not allow selecting curved regions on images. That means it is not possible to correctly analyze gel lanes if they happen to be curved. One of the requirements was to implement this functionality in our application. It is also not possible to save any data, only to export the results to comma-separated values (CSV) file. Therefore scientists have to start their work all over again every time they want to analyze a new gel.

2.3.2 Image Lab

Image Lab is a software meant for imaging and analyzing gels and blots. It supports most of the same features as ImageJ, most notably adding the possibility to analyze curved lanes. However, its main drawback is that it is not possible to save data to resume the work later. Only data export is provided [10].

2.4 Project Management

Development work was organized in one-week-long sprint cycles. Meetings with the clients and supervisor took place on Mondays and focused mostly on verifying the implementations, prioritization, and acquiring domain knowledge. Planning meetings with the developers took place on Fridays (and later on Sundays), where issues were detailed and development times estimated. For estimations the average hours suggested by all developers was used with the addition of buffer time which was modified based on the performance of prior sprints. During the final few sprints, less effort was placed on time estimations since it was clear there was no time where to split or postpone their delivery.

Summaries of the meetings, as well as various documentation can be found on the GitLab Wiki page[3]. Most of the meetings and communication took place over Microsoft Teams[4].

Milestones, issues, and source code was managed as a project on the university GitLab[5]. Feature development was done on separate branches and required approval from a reviewer before merging to the main branch.

[3]https://gitlab.cs.ttu.ee/jakutt/iaib/-/wikis/home
[4]https://www.microsoft.com/en-us/microsoft-teams/group-chat-software
[5]https://gitlab.cs.ttu.ee/jakutt/iaib

3. Project Design

This chapter will cover the frameworks and libraries used in the project, architectural choices made and the theoretical background for the method used to acquire user feedback.

The software architectural diagram is provided here for a general overview.

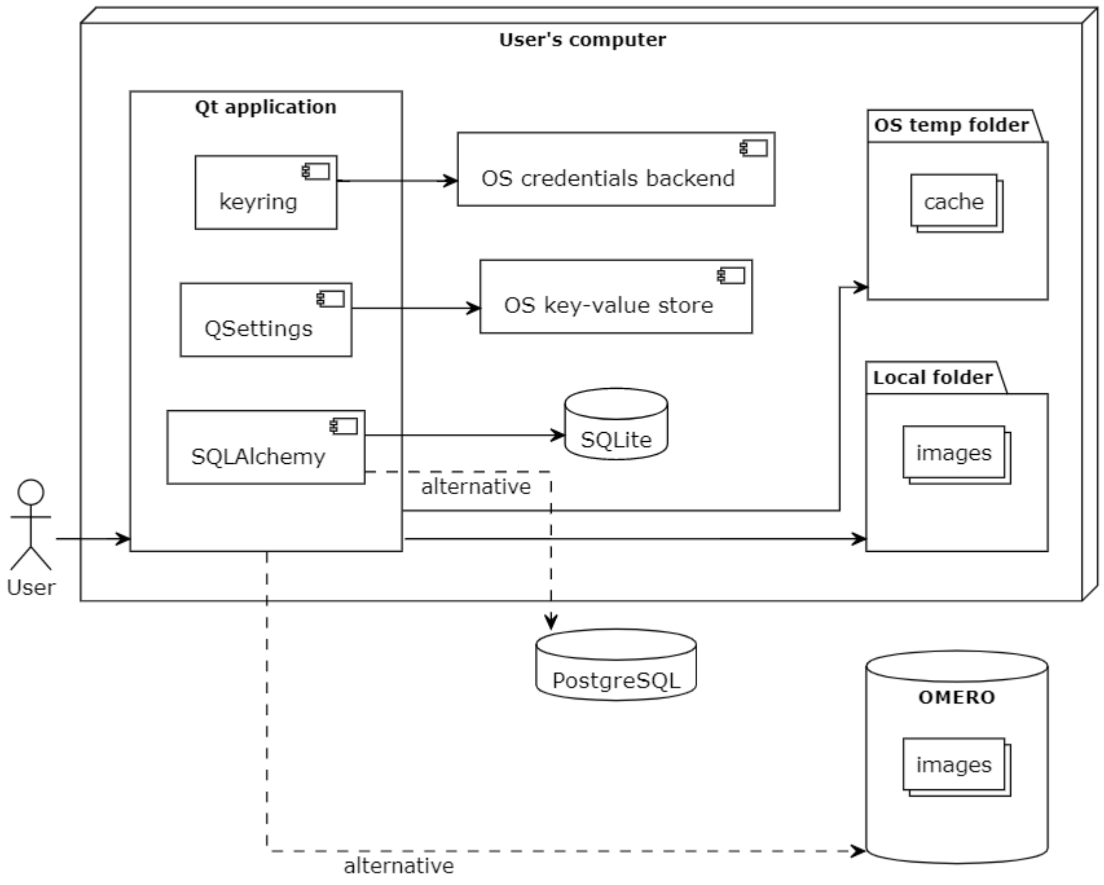


Figure 2. *Software design.*

3.1 Frameworks and Libraries

PySide6

PySide6 is the framework used to develop the application. It is the official module for the Qt for Python project. It offers all of the UI elements and functionality provided by the Qt framework to be used in Python development [11].

As an alternative, the PyQt [1] library was initially considered due to being around longer and having more usage examples available online. After short testing with both PySide6 and PyQt components, it was apparent that their application programming interfaces (APIs) were similar enough to be mostly interchangeable. PySide6 was preferred as the officially supported framework with the knowledge that a larger example base was still available.

PyQtGraph

PyQtGraph is a graphics and graphical user interface (GUI) library intended for use in mathematical, scientific, and engineering applications [12]. It is used in the project for selecting and visualizing ROIs on the Gel images, plotting intensity data, and manipulating limits for intensity analysis. It was chosen because of its high compatibility with the Qt framework, as well as for its selection of available components.

NumPy

NumPy is the fundamental package for scientific computing in Python. It ... provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays ... [13]. In the project, it is mainly used for manipulating the matrices containing the image data.

PostgreSQL

PostgreSQL is one of the most widely used object-relational database management systems [14]. In the application, it can be configured by the user to store the application data. Support for PostgreSQL integration was required by the client so they could integrate the analysis results with in-house software directly. Supporting alternative databases was a consideration and to that end, effort was made to simplify that process.

Step	**File location**
Implement adapter interface	iocbio/gel/gui/dialogs/database_connection_settings/ database_connection_settings.py
Add adapter to form	iocbio/gel/gui/dialogs/database_connection_settings/ db_selection_form.py
Add driver as a dependency	setup.py

Table 2. Steps to add alternative database support

SQLite

SQLite is a file-based Structured Query Language (SQL) engine for storing application data. It is a convenient option for storage due to its stand-alone nature and minimal configuration (path to file). The user can choose this option when locally stored data is

[1]https://www.riverbankcomputing.com/software/pyqt/

acceptable. [15]

SQLAlchemy

SQLAlchemy is the Python SQL toolkit and Object-relational mapper (ORM) that gives application developers the full power and flexibility of SQL. It provides a full suite of well known enterprise-level persistence patterns, designed for efficient and high-performing database access, adapted into a simple and Pythonic domain language [16]. It was chosen for the project for its popularity, extensive documentation and as a way to abstract the database layer for working with both PostgreSQL and SQLite.

OMERO

OMERO provides a complete platform for managing images in a secure central repository. It provides access to the images through a desktop app (Windows, Mac or Linux), from the web or from 3rd party software [17]. Providing integration for importing Gel images from OMERO was one of the main requirements for the project. Communication with the OMERO server is handled by the OMERO.py [2] client library.

Alembic

Alembic is a lightweight database migration tool for usage with the SQLAlchemy Database Toolkit for Python [18]. This tool was chosen because of its compatibility with SQLAlchemy and ease of setup.

Dependency Injector

Dependency Injector is a dependency injection framework for Python [19]. An alternative framework, python-inject[3], was considered, but Dependency Injector was found to be more widely used and has better documentation.

3.2 Architectural Choices

3.2.1 Dependency Injection

Dependency Injection is a pattern that separates service configuration from the use of services within an application. It can help make it easier to see what the dependencies are and simplify testing by allowing to replace the real service implementations with stubs or mocks [20].

[2]https://github.com/ome/omero-py

[3]https://github.com/ivankorobkov/python-inject

Python applications rarely make explicit use of the pattern as the interpreted and dynamic nature of the language makes it unnecessary [21].

However, as the project advanced and the number of components which depended on the same services grew, it became evident that using a dedicated framework for managing dependencies would help to reduce the growing complexity. Dependency Injector framework was chosen and subsequently all of the component configurations were moved into a single Container class. That Container is responsible for injecting the object instances, factories, as well as configuration parameters to the components using constructor injection.

3.2.2 Events

Qt has a concept known as signals and slots. Signals are emitted by certain actions, for example, by clicking a button. Slots are functions that listen for emitted signals and perform appropriate tasks [22].

Project software consists of various isolated views and components that nevertheless need to communicate with each other to keep different parts of the application synchronized. In the application events are managed by EventRegistry class which contains the definitions of all used signals. Every component that needs to emit or consume events could do it only through the injected EventRegistry instance. The main goals of this approach were to decrease the coupling between components, keep all the events defined in one place, and avoid the possibility of widgets emitting events that no other component of the system was listening to. It also helped to avoid the situation in which multiple different signals are actually signaling the same event.

3.2.3 Undo/redo

The software user is expected to perform a series of graph manipulations with the mouse during the intensity analysis. To avoid errors from accidental motions, the undo/redo mechanism was implemented using the Command pattern.

„The Command pattern lets toolkit objects make requests of unspecified application objects by turning the request itself into an object. This object can be stored and passed around like other objects. Use the The Command's Execute operation can store state for reversing its effects in the command itself. The Command interface must have an added Unexecute operation that reverses the effects of a previous call to Execute. Executed commands are stored in a history list. Unlimited-level undo and redo is achieved by traversing this list backwards and forwards calling Unexecute and Execute, respectively" [23].

Data modifications were executed through a single HistoryManager instance, which stored the command objects for the current user scope. History was cleared when the user changed scope (either by navigating or switching application into viewing mode) to avoid accidentally executing events which were not visible anymore. Keeping the history through deletion and creation of objects was achieved using soft-delete flags and only performing the actual deletion when the scope changed.

3.2.4 Database

The application uses one database table for keeping track of the software migration versions and eight tables for storing the data which the user either entered directly in a form field or created with graph manipulation tools. Although most of the database tables and used types were a result of the initial requirements analysis and directly reflected the needs of scientific data, some choices deserve closer attention.

Originally the *gel_image* table was supposed to store both the values of its *original_file* and *omero_id* in a single field *file_location* since the application itself doesn't have a use for that separation on the database level. The client requested to store the *omero_id* separately so they could use that in conjunction with their in-house software.

All of the tables which are directly affected by the user have a boolean *is_deleted* field to support the undo/redo functionality.

Since SQLite stores numbers as floating-point numbers [24], some of the numerical field types were also converted to floats instead of decimals. The loss in accuracy was estimated not to be an issue by the client. Fields in question are: 1) *gel_lane.protein_weight*, 2) *gel_image.rotation*, 3) *measurement_lane.value*.

Fields storing relative locations of graphical elements used internally by the software are serialized and stored as text types on the database - since their values don't hold any meaning outside of the application context, there would be no benefit from using more complex field types like JSON. These fields are: 1) *gel_image.region*, 2) *gel_image_lane.region*, 3) *gel_image_lane.zero_line_points*.

The schema of the database is shown in Figure 25 of Appendix 6.

3.3 Measuring User Confidence

During the active development of the project, emphasis was put on continuously discussing and verifying the algorithmic correctness of the required features. When the project started to fall behind in scope, features supporting the main gel image analysis flow were prioritized and tested by the client. But a lot of the features described to improve user interface and enhance user experience were discarded. By the end of the project, the authors were left with a question: the application is functional - is it also usable?

Corno et al. (2015) reviewed existing literature on how the role of the user is considered during software design and evaluation. A concept they paid closer attention to was the level of confidence the user had while interacting with technology. They note that user confidence is influenced by a correct and reliable system behavior. “One aspect of reliability, often neglected, is guaranteeing the consistency between system operation and user expectations, so that users may build confidence over the correct behavior of the system and its reaction to their actions” [25].

Since the number of people able to test our application was expected to be low and limited only to the available Systems Biology lab workers, no existing study methods were considered.

To gain a better understanding of how the application met both the testers’ expectations and user experience a Google Forms questionnaire was created. First, they were asked to identify their role by indicating whether they planned to use the software only while working at the lab or not. Then they were asked to indicate how confident they felt that the features listed worked and met their expectations. A scale of one (*Not at all confident*) to five (*Very confident*) was offered to allow the users to express their opinions more dynamically. The list of features covered in the questionnaire was derived from the user stories previously described in tables 3 - 6. Questions covered only the features that were implemented at the time of user testing.

4. Software Description

The project was implemented as a desktop Python application capable of running on Windows and Unix-like operating systems.

The application uses a SQL database to store both the operational and user-created data. If the user does not specify connection parameters for an external database the application creates a SQLite database file in the current working directory.

The application creates a cache directory under operating system (OS) provided temporary path. On Unix/Linux systems this is the path specified in the TMPDIR environment variable or */tmp* if TMPDIR is not defined. On Windows this is usually the path specified in the TEMP or TMP environment variable [26]. Two types of files are written there: images fetched from OMERO and the results of image processing steps.

Credentials necessary to authenticate with the databases are stored using the Python keyring library. Therefore, on Windows the credentials are stored to Credential Locker, on macOS to Keychain and on UNIX-like systems to KWallet [27].

4.1 Prerequisites and Installation

The minimum required Python version to use the application is Python 3.8. It is recommended to use the Python virtual environment for installing and running the application (in the following steps, the virtualenv module[1] is used). Due to an underlying dependency and the specific OS version the application is being installed on, an OS-specific C++ compiler may be required.

On Linux/Unix and macOS systems, the environment can be set up by running the following commands in the application root directory using a Unix shell program like Bash:

```
python -m virtualenv .venv
source .venv/bin/activate
```

[1]https://pypi.org/project/virtualenv/

The application can then be installed by running the following command:

```
python -m pip install -r requirements.txt .
```

If previous steps produced no errors, then application can be executed by running the following command:

```
.venv/bin/iocbio-gel
```

Installing and running the application on Windows is similar to the case of Linux. Commands can be run in Powershell or a similar command-line application. Setting up the application:

```
py -m virtualenv .venv
.venv\Scripts\activate
```

Installing the application:

```
py -m pip install -r requirements.txt .
```

Running the application:

```
.venv\Scripts\iocbio-gel
```

4.1.1 Application Startup

During the initial startup, the user is shown a database configuration dialog where it's possible to specify the database management system (DBMS) to be used and the parameters to connect to the database, as well as an image source configuration dialog, where the user can choose between local filesystem storage or specify connection parameters to an OMERO server. The application tries to connect with given parameters and shows an error message if a connection can not be established. The user can then add the correct parameters. If the user closes the dialog, then the application does not proceed and is closed.

After the database connection has been established SQLAlchemy checks the migration history. Any unapplied migration scripts located at *iocbio/gel/db/alembic/versions/*.py* will be run to ensure that the database schema conforms to the requirements of the application.

When the application is configured to connect to an OMERO instance, then, during the startup process, fresh copies of images are fetched from OMERO and stored in the local cache folder.

4.2 Application Layout

Visually the application is divided into three components: mostly static toolbar on top, a narrow navigation menu with embedded gels list on the left, and the main content area.

4.2.1 Toolbar

The horizontal toolbar at the top of the application provides the user with access to both context-dependent and globally relevant functionalities. “Undo” and “Redo” buttons allow the user to revert their previous action or reapply previously reverted actions. “Current mode: Editing” and “Current mode: Viewing” either enable or disable the user’s ability to modify objects and data. The “Settings” button acts as a link to the settings view. “Add new Gel” or “Add new Measurement” buttons are available depending on the view and provide a shortcut for creating new objects.

4.2.2 Navigation

Left side of the application contains a “Gels” button which opens up the gels list view, a list of individually named gel buttons which lead the user to gel detail view for the selected gel and a “Types” button at the bottom, which opens the measurement types view.

4.2.3 Content Area

The main area which displays the views listed below. On the application startup the gels list view is displayed by default.

4.3 Application Views

The application views can be grouped into three different categories:

- data tables which provide general overview of previous work and allow to specify context for the analysis - gels list view, gel detail view, measurement types view;
- gel image and graph manipulation views for the analysis steps - raw, adjust, background, lanes and measurements;
- settings view for changing the application configurations.

4.3.1 Gel List View

A table view where the user can navigate by clicking on the “Gels” button on the left upper corner of the application. The following columns are presented for every gel: ID, name, transfer, comment. In addition, the counts of attached lanes and measurements are shown. The value of every field except for the ID can be changed if the editing mode is active. The gel list view can be seen in Figure 14.

4.3.2 Gel Detail View

To navigate into the gel detail view one must click on one of the buttons with gel names on the left menu below the “Gels” button. On the upper part of the view, the gel name and creation time are displayed. The lanes table contains the ID, lane number, sample ID, protein amount in micrograms (µg), comment fields, and a checkbox field indicating if it is a reference lane that samples can be compared to. All fields except the ID can be changed if the editing mode is turned on. Lanes can also be deleted in this mode. The measurements table contains the data of all the measurements linked to the gel. The table shows the image, time, ID, type, comment, and identifiers of connected lanes. All the fields can be changed in editing mode. The gel detail view can be seen in Figure 17.

4.3.3 Measurement Types View

The measurements type view is a table view where one can navigate by clicking the “Types” button in the lower-left corner, below the gel buttons. The table shows the ID, name, and comment of every measurement type. The value of each field can be changed and the type can be deleted in editing mode. An error box appears and deletion fails if the user tries to delete a measurement type that is connected to one or more measurements. The measurement types view can be seen in Figure 16.

4.3.4 Gel Image Raw View

The gel image view contains different sub-views meant for processing and analyzing the gel image. The user can move between these views by clicking on corresponding tabs. The raw image view shows the gel image without any processing done on it. Image can not be changed in this view.

4.3.5 Gel Image Adjust View

Adjust view allows the user to specify a region on the image that is used for analysis in subsequent steps. A rectangle appears when the image is clicked. The size, rotation, and position of this rectangle can be changed by dragging it from the handles. The region can be confirmed for analysis by clicking on the "Apply" button. Adjust view can be seen in Figure 20.

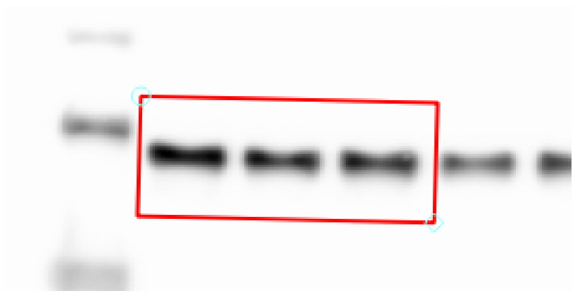


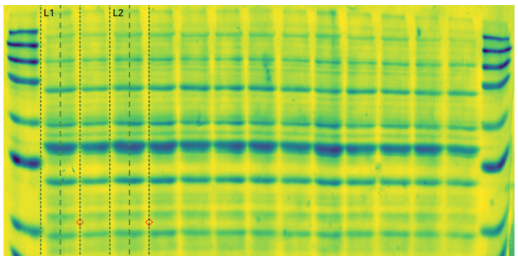
Figure 3. *Cropping/rotation selection ROI.*

4.3.6 Gel Image Background View

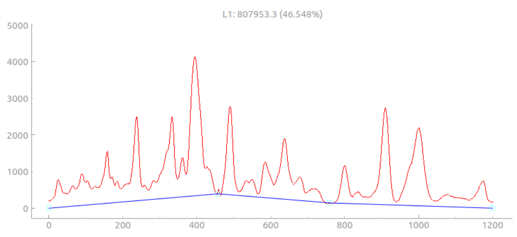
The background subtraction view gives the user the option to apply the rolling ball background subtraction algorithm, specifying kernel type ("ball", "ellipsoid", or "none"), ball radius (two radiuses for ellipsoid), and whether the image is inverted before passing into the algorithm. When "none" is chosen, background subtraction will not be performed). Pressing "Apply" will execute the algorithm with the given parameters. Three images are shown: how it looked from the previous step, the extracted background from the algorithm, and the result, which is the difference of the input image and background (matrix subtraction). If the image was set to be inverted, the algorithm will use an inverted version of the image and the result will be subtracted from the inverted image, which will then be inverted again to restore the original colors. The image should be inverted if the source image has a dark background. The rolling ball implementation used is the one from the scikit-image library [28]. The gel image background view can be seen in Figure 21.

4.3.7 Gel Image Lanes View

The lanes view contains an image graph with the image processed in the three previous views. The "New lane" button allows the user to place a vertical variable-width lane inside the bounds of the image, which will span from top to bottom. Existing lanes can be moved along the horizontal axis, change widths, or be removed entirely. The number of placeable lanes is bounded by the number of lanes defined in the gel detail view. For each lane, there is a corresponding graph showing the pixel intensities along the lane, the y-axis being the pixel intensity and the x-axis the position of the horizontal chunk of pixels in the lane, starting from the top of the lane. The graph has a modifiable zero-line below the reading that can be dragged around by their points, which can be added to the line by clicking on it. The resulting area for the lane is determined by the difference between the readings integral and zero-line integral from zero to the height of the lane. The gel image lanes view can be seen in Figure 22.



(a) Gel image lanes view: image plot with lanes L1 and L2.



(b) Gel image lanes view: graph for lane L1 zero-line in blue.

Figure 4. Gel image graph and lanes plot

4.3.8 Gel Image Measurements View

The measurements view contains four components: a) lanes' graph, b) intensity plots, c) measurement lanes table, and d) measurements table. (Figure 23)

a) Image graph with the lanes marked on the previews analysis step. Placed lanes can't be manipulated in this view anymore. Clicking on a lane allows the user to add or remove it from the selected measurement.

b) An intensity plot is shown for every selected lane for the current measurement. Each plot shows the vertically combined pixel intensity values for every pixel row along the lane. Minimum and maximum integration limit lines can be adjusted to specify what area of the plot should count toward the area of interest.

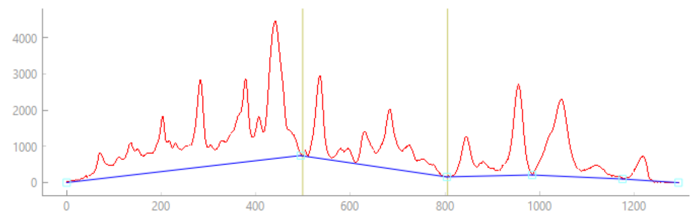


Figure 5. *Integration limit lines*
Shown as vertical yellow lines

c) An optional comment can be added to the measured area per lane and indicated whether the measurement was successful or not.

d) Measurement type and comment can be added. Clicking on a measurement row will mark that measurement as selected and update the information on other components to match that change.

4.3.9 Settings View

The user can navigate to the settings view by clicking on the “Settings” button on the toolbar. It allows the user to change configured database settings and offers the same options as the initial database dialog. The “Image source” section allows the user to choose between local and OMERO options for storing images. The “Database connection” section lets users specify the same database connection options as in the database configuration dialog. The settings view can be seen in Figure 24.

Database Configuration

The database configuration section lets the user choose between SQLite and PostgreSQL for the DBMS from the dropdown menu, and according to the choice specific configuration parameters will appear. In the case of SQLite, there are no additional settings to be shown. For PostgreSQL, the user can specify database host, port, Secure Sockets Layer (SSL) mode, database name, user, and password. This section will be shown to the user as a dialog when the application is first started and no database is configured.

Image Source Configuration

The image source configuration section lets the user choose the source for the images to be analyzed. The user can choose between a folder on the local filesystem and an existing OMERO database. If the local filesystem is chosen, files will be fetched from a folder in the application’s directory. If OMERO is selected, the user can specify the hostname, port, username, and password for the connection. This section is shown to the user as a dialog when the application is started and the image source has not been configured.

5. Validation

To validate if the goals set out by the project were successful, we rely on both comparing the list of features initially described as requirements and the results of user feedback testing.

28 out of the 39 identified user stories were implemented during the project. 27 of those were included in the user feedback questionnaire - one was excluded since it was only a requirement specific to the project lead at the Systems Biology lab - "I want the software to be released under GPLv3 license". The questionnaire was filled out by three of the lab workers.

Summary of user feedback after testing:

- testers felt very confident that 16 features worked and met their expectations;
- all but one feature - "I can use the software on macOS computer" - got at the highest confidence rating from at least one tester;
- average rating on a scale of one to five was 4.65.

The full result of the feedback can be seen in Table 9 of Appendix 5.

General feedback from the client was that the software can perform the main use cases regardless of the features which were not finished. It can be used to manage the process and perform intensity analysis on gel images. Thus the primary project goal set during the start of the project was achieved.

6. Results

Despite not meeting all the requirements set out at the start of the project, the authors were able to prioritize features supporting the primary goal of creating software that would reduce the gel analysis process time for the Systems Biology lab. The application allows a scientist to prepare and manage the gel context for analyses, measure protein sample quantities on the gel images and review prior work. The authors gained new knowledge and skills about used technologies, the development process, and the domain of the project.

An overview of which of the requirements were implemented can be seen in tables 3 - 6 of Appendix 2.

Seven features that were part of the requirements for the secondary goal of making the software publicly available as an open-source project were not implemented:

- I want GitLab pipeline configured to build packaged executables;
- I want the software to look as agreed on the UI design document;
- I want to use the software in my own language;
- I want to export the results of my work as a CSV file;
- I want to curve the lanes individually;
- I want a user manual for the software;
- I want the implementation of analysis steps formally documented.

6.1 Internationalization

At the ending phases of development, clients communicated the requirement for application to support other languages besides English. That would have been necessary for users whose first language is not English. PySide has a package for the convenient implementation of internationalization [29], and therefore authors believe that adding such functionality will not be a big issue in the future. It was currently left out of the scope mainly because it was communicated too late.

6.2 CSV Export

Exporting gel and measurement data to Excel would have been useful for performing additional analysis in third-party software. Since it was not a high priority requirement, it was left out of the project scope due to time constraints. It should not be an overly complicated feature to add in the future, as generating CSV files from relational data is a relatively standard task.

6.3 Curved Lanes

The most prominent feature that was not implemented was the ability to select curved lanes for analysis. According to the clients, the need for that feature would be rare and only required when the physical gel gets damaged or distorted, but in that circumstance, it would be valuable. They estimated that a loss of a gel would result in two days of work to prepare another gel and that only if the study material is still available. Unfortunately, it did not get implemented as it was also one of the most complicated features.

6.4 Gel Filtering

The possibility to filter the gel list by search queries would be useful if there are many gels added. This would have also improved the user experience to a large degree. It was decided to leave it unimplemented due to it not being critical for application functioning and having more important features to implement at the time.

7. Comments and Discussion

This section provides analysis of the development process and an overview of important elements in the technology stack, with commentary on specific aspects during development that involved significant decisions or affected flow of development.

7.1 Domain Knowledge

At the time of choosing this topic for the thesis, we had little knowledge about protein determination methods and processes, how/what applications are used to perform analysis, and why. As we started working on the project and having meetings with the project owners, we received some details on the current way of processing gel images and their shortcomings. However, our understanding of how some features should work or relate to the processes was partially faulty for some aspects, which required some rework down the line. This may have been mitigated with more detailed communication and development of clearer specifications, and by periodically releasing partial versions of the applications for the project owner to test specific aspects of workflows. More thorough research into the backgrounds of the algorithms and motivations behind the methods, stepping into the scientist's shoes, may have also helped for some implementations.

7.2 Project Management

Two main causes were identified that contributed to a significant enough time delay and eventually resulted in cutting some features out of the scope - communication and external time management.

Although the features initially planned for the start of the project were simplistic, the authors spent a lot of time familiarizing themselves with the framework, libraries, and exact requirements of the project, but they did not take enough time to agree on development practices and communicate their specific implementation details before developing those features. This resulted in an initial codebase where duplication was hard to spot and rectify because of three varying styles of implementation - most of which required refactoring later on.

It was an open discussion on whether or not the authors should have tried to more specialize in developing different aspects of the application. This would have perhaps decreased the combined time that was spent on individually learning all the frameworks and libraries. It would have also forced a stricter initial structure for the software and exposed the need to improve communication between the authors. Unfortunately, due to circumstances, the authors were able to work on the project on differing days of the week, and the communication overhead felt more like a blocker. Additionally, the initial development effort was placed on creating the different application views, and so at the time it made sense to focus everyone on creating vertical slices of the software.

Secondly, external time constraints were not properly taken into account in project management during the second half of the project. The authors were too optimistic in estimating their capabilities or available time and realized fairly late that the project scope should be reprioritized.

The project management process failed to properly identify which of the external time sinks were recurring in their nature and which were one-time events. This made planning issues for upcoming sprints more difficult. A strategy to cope with this was adopted in form of splitting the issues even smaller so it would be possible to pick them up more readily in an ad-hoc manner. It helped a little but was not a sufficient solution. When a developer fell behind on their input, the situation was compounded by the additional need to familiarize themselves with changes done in the software in the meanwhile. As a remedy, perhaps implementing pair-programming sessions would have helped, but the need for this was not communicated clearly at the time.

7.3 Curved Lanes

One of the initial requirements for the application, curved lanes, was eventually found to be notably difficult to implement, and therefore not implemented within the current scope. There was one idea of how it may be done relatively easily, using the MultiRectROI widget. It was a ready-made widget that allowed chaining together multiple rectangular ROIs at variable angles and would've provided a single concatenated array of image data, much like a regular ROI for easy usage [30]. The issue with it was that the ROIs when connected at angles, overlapped with each other at one side and left a gap on the other, which wouldn't have been the desired result. Other ideas revolved around creating polygonal ROIs that joined edge-to-edge and could have curved edges or simply two curved lines (PyQtGraph offers methods to implement curved lines using the PlotCurveItem class [31]) from which the region data would be calculated with a custom-made algorithm, staying perpendicular to the centerline. Both of those ideas would have needed to solve the issue of curving causing one side to have fewer pixels than the other, meaning each rectangular row of pixels had to expand into a circular arc from the shorter line to and along with the longer. We had little idea of how to properly/most efficiently implement this and feared it would eat up too much time, so in the face of time constraints, we decided to drop it.

7.4 Qt

Using the Qt framework for our application was considered a given since it was desired by the project owner and deemed standard for the problem domain and user base. Since we were unfamiliar with the framework and making standard desktop applications, the rate of feature development was affected. It was difficult to tell if opting for a more familiar web-based app paradigm would've decreased initial friction and allowed us to implement more features, as some of the libraries for image processing and plotting might've been hard to find mature and feature-equivalent versions for. Performance implications were also considered, as some image processing algorithms are slow even in the current implementation. Visuals of the application may have been significantly better though, as Hypertext Markup Language (HTML) / Cascading Style Sheets (CSS) pages are easier to customize.

7.5 Python

We had the option to either use C++ or Python with a wrapper library for Qt, and we chose the latter due to familiarity and perceived ease of development when compared to the former. While the application might've been more performant on C++, the extra layer of unfamiliarity with the language, its lack of memory management, and fewer batteries-included libraries with usable widgets like PyQtGraph and skimage would have most likely increased the learning curve and development time for the same set of features significantly. The overall performance of the application, even with the overhead of Python, is good, with one exception being the background subtraction algorithm, which is implemented differently from its ImageJ equivalent.

7.6 Application Architecture

During the later stages of development, the authors found out about Qt's Model-View[1] system, which may have benefited the project significantly. Having a separate model object to more conveniently map the database entities and allow for easy state sharing across view components may have made implementing and maintaining some aspects,

[1]https://doc.qt.io/qt-6/model-view-programming.html

like Create, Read, Update, Delete (CRUD) tables for measurements and lanes, easier. The documentation for it wasn't very in-depth, however, so the question of how to properly encode more complex views like graphs and integrate with other libraries remained and may have been too much of an undertaking by that stage of development.

7.7 Alternative Image Sources

Not nearly enough was done to support alternative image sources, as it was the case for adding databases. There is an interface *select_image.py*[2] for implementing the visual dialog component which is shown to the user when attempting to select or change an image. But, since fetching images from a local folder didn't require managing a connection, a common interface for such a client was not created. As such, the first steps for implementing a new image source would be to follow the implementation and usage of the *omero_client.py*[3] class and separate image source specifics from the shared interface.

[2]iocbio/gel/gui/dialogs/select_image.py
[3]iocbio/gel/application/image_source/omero_client.py

8. Conclusion and Future Development

The project met its primary goal - the created application aids biologists sufficiently to be usable in laboratories for managing analysis process metadata and measuring the quantities of the protein samples on the gel images. During the project, 28 out of the 39 identified user stories were implemented, and the overall feedback from the client and the testers was positive.

The requirements for the secondary goal of being able to publish the software as an open-source project were not met due to time constraints. Some development was done towards that goal, but not all of the features were implemented or were implemented in a limited way.

In addition to finishing the missed functionality, there are some features that the authors believe could enhance the utility of the application.

Certain processes like lane and ROI detection, zero-line placement, and background detection could at least to some extent be automated. Zero-line placement should be relatively straightforward to implement as it is only necessary to determine the minimum value of the data and draw the horizontal line accordingly.

Currently, background subtraction is slow, partly because of the image dimensions. A possible option to reduce the time to run the rolling ball algorithm would be to downscale the image, apply the algorithm to get the background matrix, and then upscale the image again, and then subtract that from the original. The process could also be optimized by performing the background subtraction on a separate thread and avoiding freezing the UI. This way, the user could also stop the process if necessary.

Bibliography

[1] *Protein Quantification Using ImageJ*. https://openwetware.org/wiki/Protein_Quantification_Using_ImageJ. Accessed: 2022-05-08.

[2] Maria Hayes. “Measuring Protein Content in Food: An Overview of Methods”. In: *Foods* 9.10 (2020). ISSN: 2304-8158. DOI: 10.3390/foods9101340. URL: https://www.mdpi.com/2304-8158/9/10/1340.

[3] Sebastian Seiffert. *Supramolecular Polymer Networks and Gels*. Vol. 268. Jan. 2015. ISBN: 978-3-319-15403-9. DOI: 10.1007/978-3-319-15404-6.

[4] Sameh Magdeldin, ed. *Gel Electrophoresis - Principles and Basics*. 2012. DOI: 10.5772/2205. URL: https://app.dimensions.ai/details/publication/pub.1108429868.

[5] Ellen C. Jensen. “The Basics of Western Blotting”. In: *The Anatomical Record* 295.3 (2012), pp. 369–371. DOI: https://doi.org/10.1002/ar.22424. eprint: https://anatomypubs.onlinelibrary.wiley.com/doi/pdf/10.1002/ar.22424. URL: https://anatomypubs.onlinelibrary.wiley.com/doi/abs/10.1002/ar.22424.

[6] Gholam Hossein Meftahi et al. “Applications of western blot technique: From bench to bedside”. In: *Biochemistry and Molecular Biology Education* 49.4 (2021), pp. 509–517. DOI: https://doi.org/10.1002/bmb.21516. eprint: https://iubmb.onlinelibrary.wiley.com/doi/pdf/10.1002/bmb.21516. URL: https://iubmb.onlinelibrary.wiley.com/doi/abs/10.1002/bmb.21516.

[7] Axel Rosell et al. “Evaluation of different commercial antibodies for their ability to detect human and mouse tissue factor by western blotting”. In: *Research and Practice in Thrombosis and Haemostasis* 4.6 (2020), pp. 1013–1023. DOI: https://doi.org/10.1002/rth2.12363. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rth2.12363. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/rth2.12363.

[8] Josephine L. Backhouse and Serge I. Nesteroff. “Treponema pallidum western blot: Comparison with the FTA-ABS test as a confirmatory test for syphilis”. In: *Diagnostic Microbiology and Infectious Disease* 39.1 (2001), pp. 9–14. ISSN: 0732-8893. DOI: https://doi.org/10.1016/S0732-8893(00)00213-3. URL: https://www.sciencedirect.com/science/article/pii/S0732889300002133.

[9] Virginia García-Solaesa and Sara Abad. “SDS-Polyacrylamide Electrophoresis and Western Blotting Applied to the Study of Asthma”. In: vol. 1434. June 2016, pp. 107–120. ISBN: 978-1-4939-3650-2. DOI: 10.1007/978-1-4939-3652-6_8.

[10] *Image Lab*. https://www.bio-rad.com/webroot/web/pdf/lsr/literature/10000076953.pdf. Accessed: 2022-05-08.

[11] *PySide6*. https://pypi.org/project/PySide6. Accessed: 2022-04-28.

[12] *PyQtGraph*. https://www.pyqtgraph.org/. Accessed: 2022-04-28.

[13] *NumPy*. https://numpy.org/doc/stable/. Accessed: 2022-04-28.

[14] *PostgreSQL*. https://www.postgresql.org/. Accessed: 2022-04-28.

[15] *SQLite*. https://www.sqlite.org/about.html. Accessed: 2022-04-28.

[16] *SQLAlchemy*. https://www.sqlalchemy.org/. Accessed: 2022-04-28.

[17] *OMERO*. https://docs.openmicroscopy.org/omero/5.6.4/users/index.html. Accessed: 2022-04-28.

[18] *Alembic*. https://alembic.sqlalchemy.org/en/latest/. Accessed: 2022-04-28.

[19] R. Mogylatov. *Dependency Injector*. https://python-dependency-injector.ets-labs.org/index.html. Accessed: 2022-05-08.

[20] M. Fowler. *Injection*. https://www.martinfowler.com/articles/injection.html. Accessed: 2022-05-08.

[21] R. Mogylatov. *Dependency injection and inversion of control in Python*. https://python-dependency-injector.ets-labs.org/introduction/di_in_python.html. Accessed: 2022-05-08.

[22] *Signals Slots*. https://doc.qt.io/qt-5/signalsandslots.html. Accessed: 2022-04-28.

[23] Johnson, R. E. Vlissides, J. Gamma, E. Helm, R. “Design patterns: Elements of reusable object-oriented software”. In: Reading, Mass: Addison-Wesley, 1995, pp. 263–266.

[24] *Floating Point Numbers*. https://www.sqlite.org/floatingpoint.html. Accessed: 2022-05-08.

[25] Fulvio Corno et al. “Designing for user confidence in intelligent environments”. In: *Journal of Reliable Intelligent Environments* 1.1 (July 2015), pp. 11–21. DOI: 10.1007/s40860-015-0001-7. URL: https://doi.org/10.1007/s40860-015-0001-7.

[26] *QDir Class*. https://doc.qt.io/qt-6/qdir.html#tempPath. Accessed: 2022-05-04.

[27] *Keyring*. https://pypi.org/project/keyring/. Accessed: 2022-05-08.

[28] *Scikit-image rolling ball algorithm*. https://scikit-image.org/docs/stable/api/skimage.restoration.html#skimage.restoration.rolling_ball. Accessed: 2022-04-28.

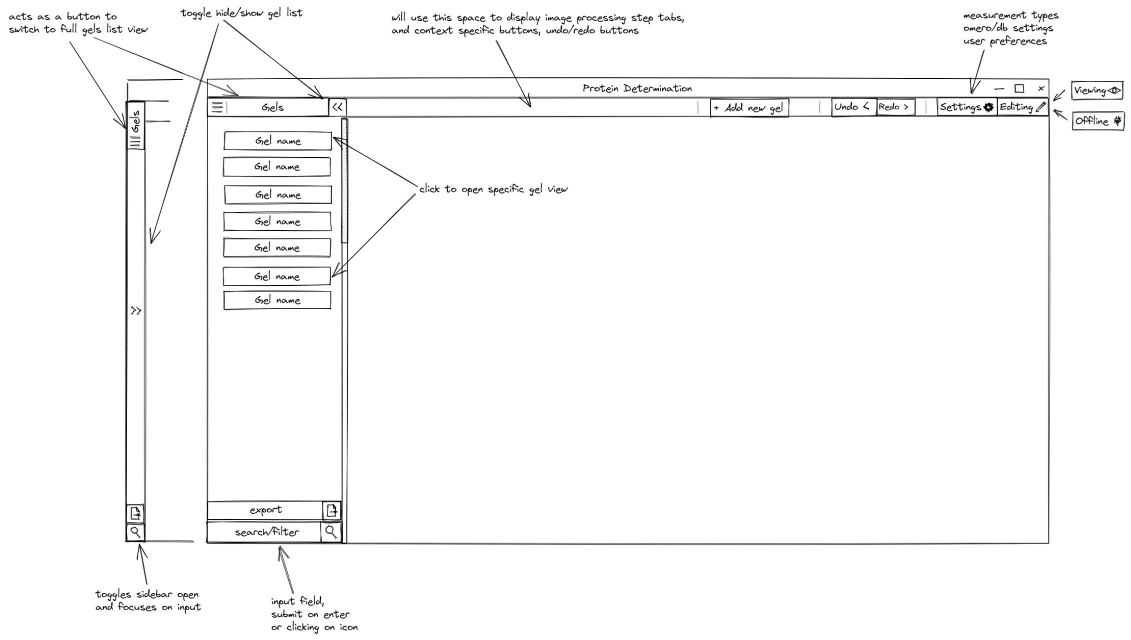
[29] *PySide internationalization*. https://wiki.qt.io/PySide_Internationalization. Accessed: 2022-05-01.

[30] *pyqtgraph.MultiRectROI*. https://pyqtgraph.readthedocs.io/en/latest/graphicsItems/roi.html#pyqtgraph.MultiRectROI. Accessed: 2022-05-09.

[31] *PlotCurveItem*. https://pyqtgraph.readthedocs.io/en/latest/graphicsItems/plotcurveitem.html. Accessed: 2022-04-28.

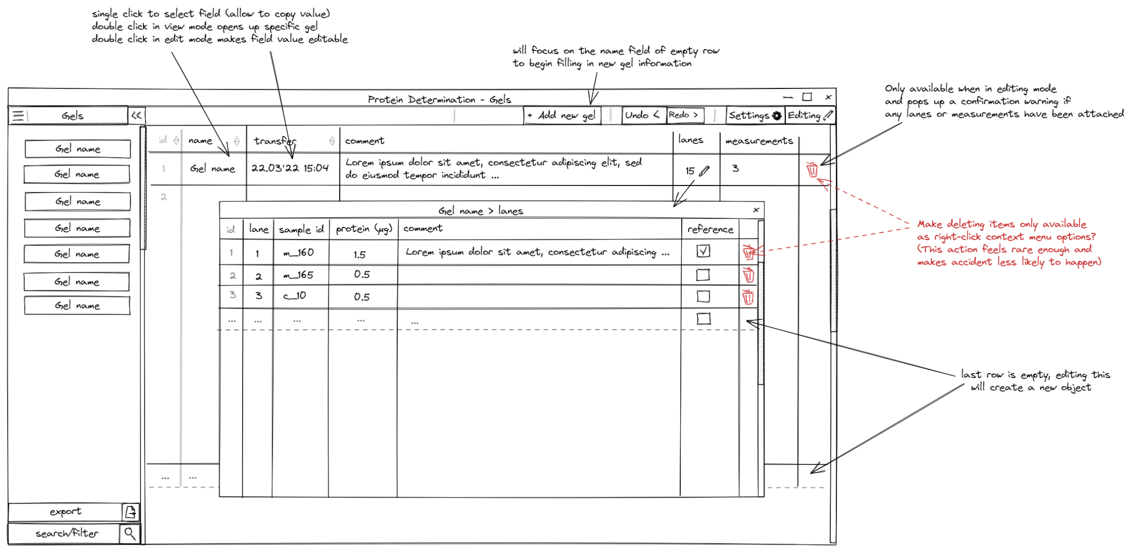
Appendix 1 – Application UI Draft

Design draft as agreed with the client as part of the requirements.



acts as a button to switch to full gels list view
toggle hide/show gel list
will use this space to display image processing step tabs, and context specific buttons, undo/redo buttons
measurement types omero/db settings user preferences
Protein Determination
Gels
+ Add new gel
Undo
Redo
Settings
Editing
Viewing
Offline
Gel name
click to open specific gel view
export
search/filter
toggles sidebar open and focuses on input
input field, submit on enter or clicking on icon

Figure 6. *Application layout.*



single click to select field (allow to copy value)
double click in view mode opens up specific gel
double click in edit mode makes field value editable
will focus on the name field of empty row to begin filling in new gel information
Only available when in editing mode and pops up a confirmation warning if any lanes or measurements have been attached
Protein Determination - Gels
Gels
+ Add new gel
Undo
Redo
Settings
Editing
Gel name
id
name
transfer
comment
lanes
measurements
22.03'22 15:04
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ...
15
3
Gel name > lanes
lane
sample id
protein (μg)
reference
m_160
m_165
c_10
1.5
0.5
Lorem ipsum dolor sit amet, consectetur adipiscing ...
Make deleting items only available as right-click context menu options? (This action feels rare enough and makes accident less likely to happen)
last row is empty, editing this will create a new object
export
search/filter

Figure 7. *Application gels list view.*

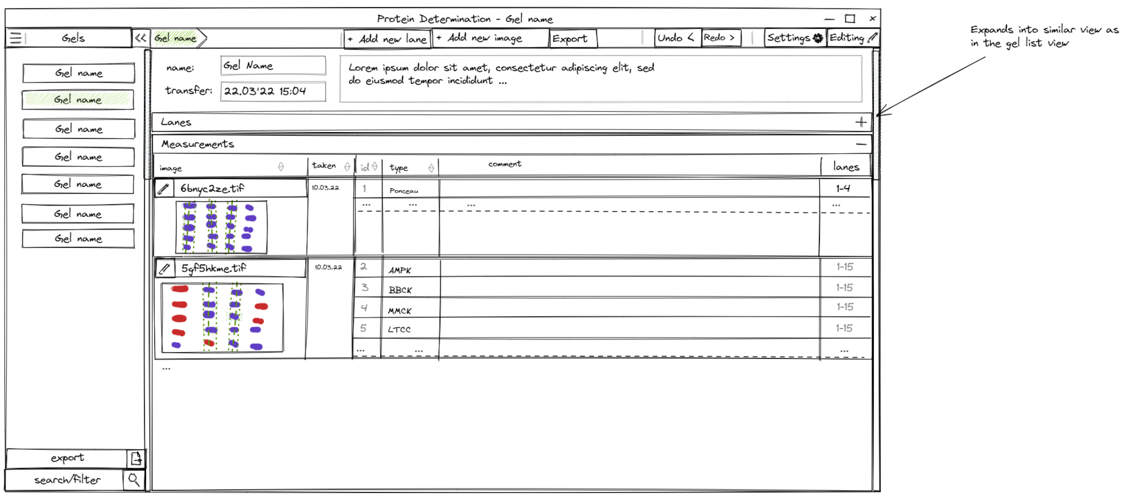


image	taken	id	type	comment	lanes
6bnyc2ze.tif	10.03.22	1	Ponceau		1-4
		...	...	...	...
5gf5hkme.tif	10.03.22	2	AMPK		1-15
		3	BBCK		1-15
		4	MMCK		1-15
		5	LTCC		1-15
		...	...		...
...					

Figure 8. *Application gels single view.*

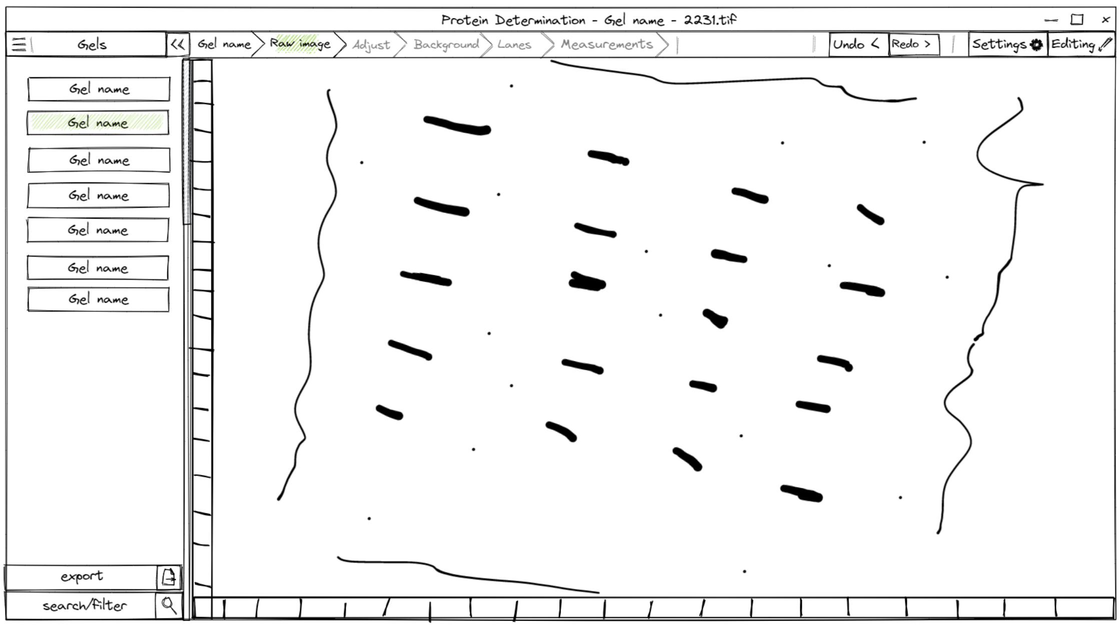


Figure 9. *Application raw image view.*

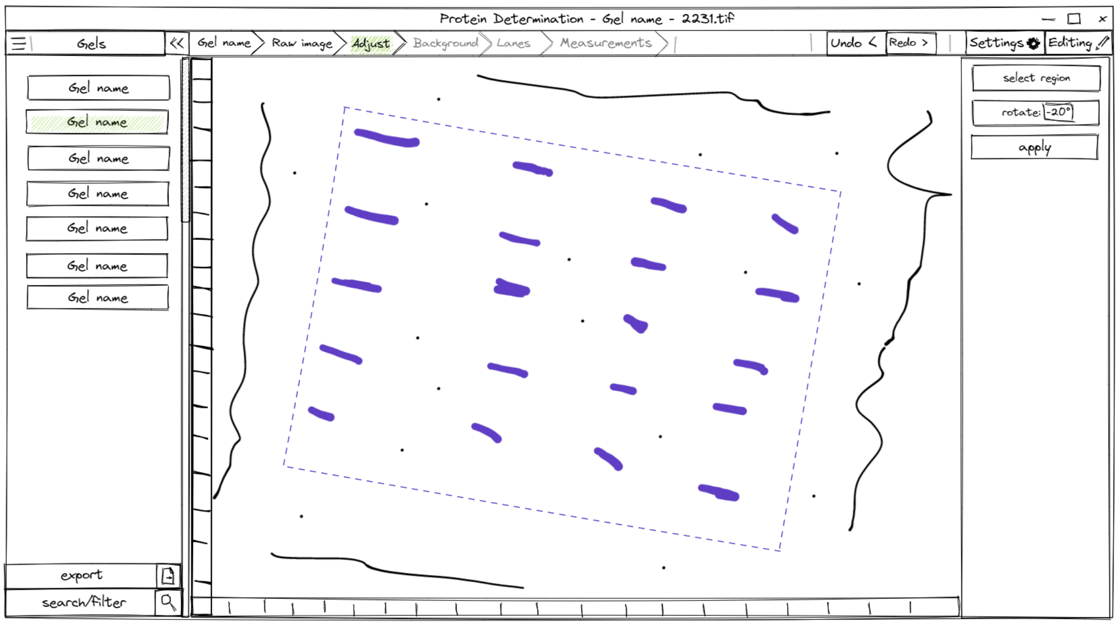


Figure 10. *Application adjust image view.*

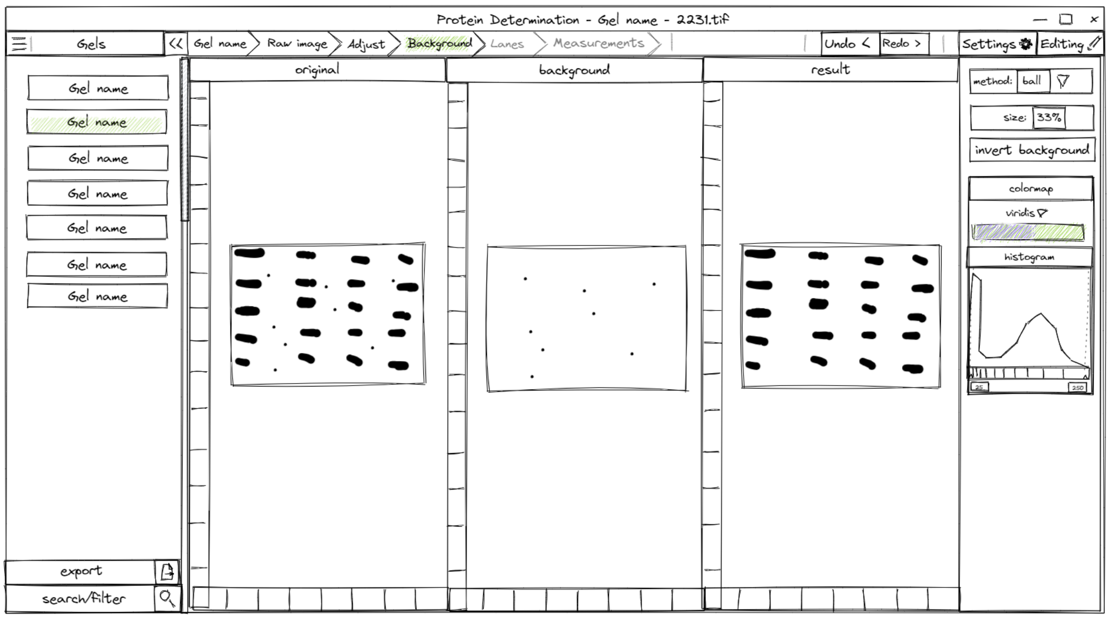


Figure 11. *Application gel background view.*

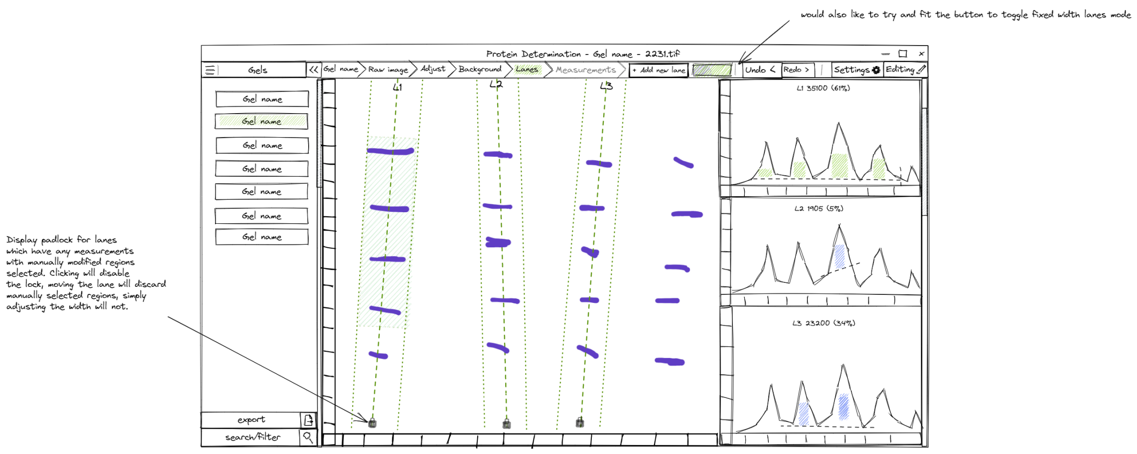
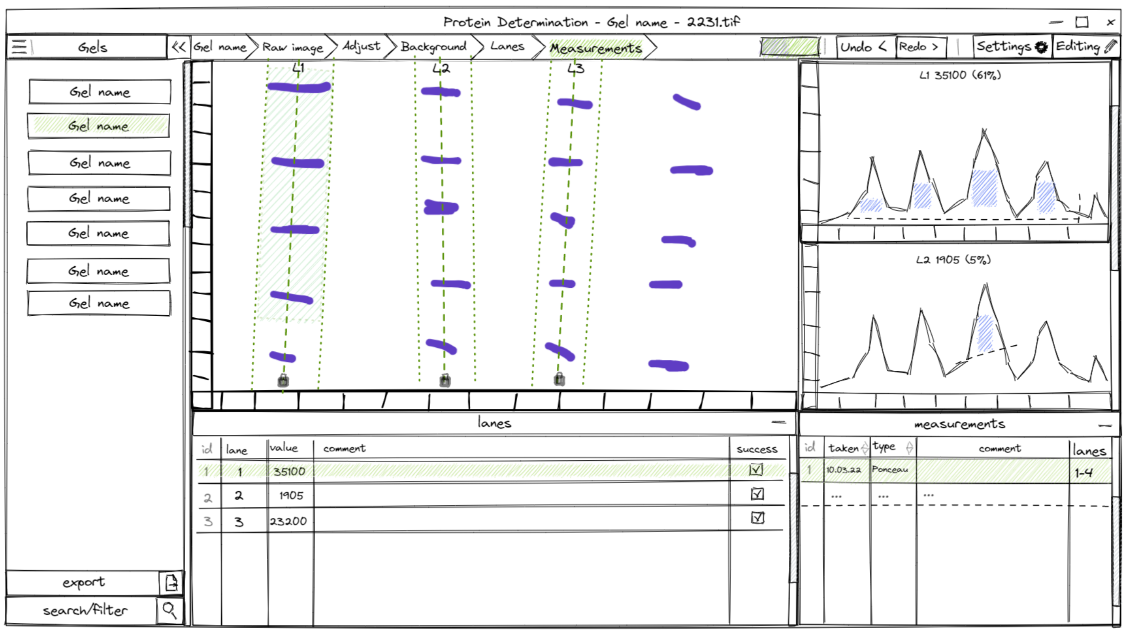

would also like to try and fit the button to toggle fixed width lanes mode
Protein Determination - Gel name - 2231.tif
Gels
Gel name
Raw image
Adjust
Background
Lanes
Measurements
+ Add new lane
Undo <
Redo >
Settings
Editing
L1
L2
L3
L1 35100 (61%)
L2 1905 (5%)
L3 23200 (34%)
Gel name
Gel name
Gel name
Gel name
Gel name
Gel name
Gel name
export
search/filter
Display padlock for lanes which have any measurements with manually modified regions selected. Clicking will disable the lock, moving the lane will discard manually selected regions, simply adjusting the width will not.

Figure 12. *Application gel lanes view.*



lanes

id	lane	value	comment	success
1	1	35100		☑
2	2	1905		☑
3	3	23200		☑

measurements

id	taken	type	comment	lanes
1	10.03.22	Ponceau		1-4
	...	...	...	

Figure 13. *Application gel measurements view.*

Appendix 2 – User Stories

Requirements described as user stories of the identified roles.

	As the Systems Biology lab project lead	**Implemented**
1	I want the software to be released under GPLv3 license.	Yes
2	I want GitLab pipeline configured to build packaged executables.	No
3	I want the software to look as agreed on the UI design document.	No

Table 3. Requirements of the Systems Biology lab project lead

	As a scientist working at the Systems Biology lab	**Implemented**
4	I want to import images from the lab's OMERO server.	Yes
5	I want my data saved on the lab's PostgreSQL server.	Yes
6	I want the software to indicate when it has lost external connections and allow to retry or change connection parameters.	No

Table 4. Requirements of scientist working at the Systems Biology lab

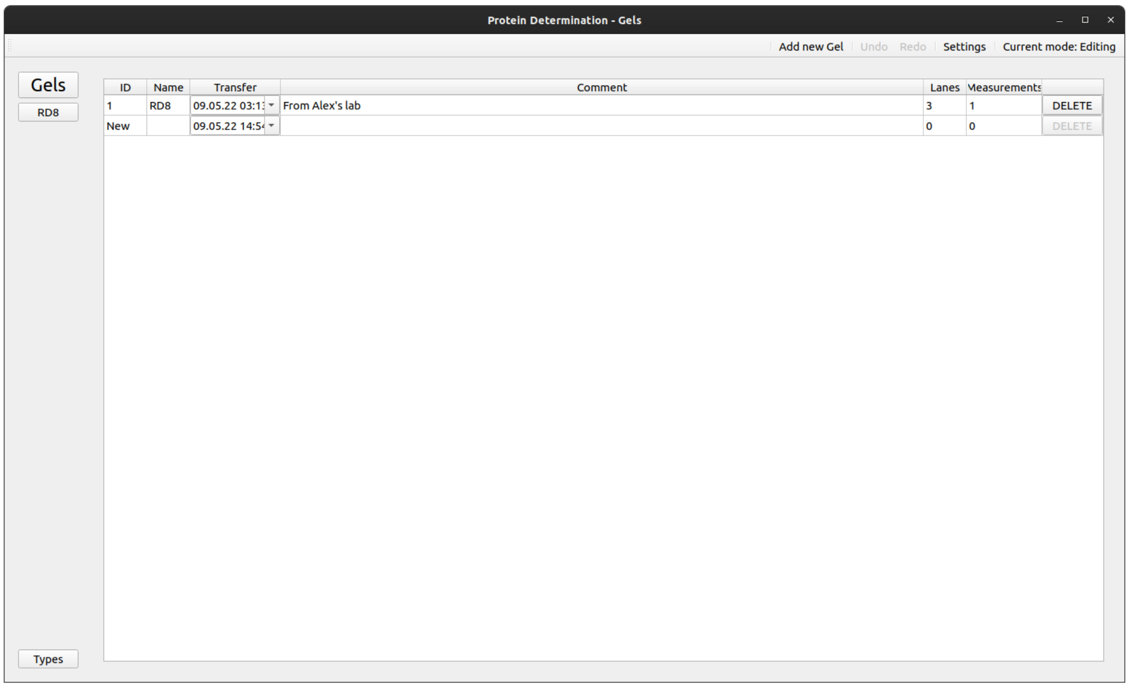
	As a scientist not working at the Systems Biology lab	**Implemented**
7	I want to import images from my local computer.	Yes
8	I want my data to persist between work sessions locally.	Yes
9	I want to export the results of my work as a CSV file.	No
10	I want to use the software in my own language.	No

Table 5. Requirements of scientist not working at the Systems Biology lab

	As a scientist	**Implemented**
11	I want to add and modify measurement types.	Yes
12	I want to add and modify gels.	Yes
13	I want to add and modify the general information about the lanes on the gel.	Yes
14	I want to add multiple images for gels.	Yes
15	I want to view the raw image before analysis.	Yes
16	I want to select a ROI on the image before analysis (rotate and crop).	Yes
17	I want to specify if the background on the image is dark or light.	Yes
18	I want to correct image background (subtraction by rolling ball method).	Yes
19	I want to choose a false-color mapping for the image.	Yes
20	I want to mark the lane positions and widths on the Gel image.	Yes
21	I want to adjust lane widths individually.	Yes
22	I want to rotate the lanes individually.	No
23	I want to curve the lanes individually.	No
24	I want to see pixel intensity plots for marked lanes.	Yes
25	I want to specify a zero-line to separate background from signal on the intensity plots.	Yes
26	I want to add and modify measurements connected to specific gel images.	Yes
27	I want to choose which lanes a measurement includes.	Yes
28	I want to specify the integration limits on the pixel intensity plots.	Yes
29	I want the selected limits to be visible on their lanes.	No
30	I want to see a comparison and annotate analyzed lanes for measurements.	Yes
31	I want to search previously analyzed gels by name, ID, sample ID, lane ID and measurement ID.	No
32	I want the option to undo an accidental modification.	Yes
33	I want a way to disable all editing operations to avoid accidental modifications during a review.	Yes
34	I want to change my image and data source parameters after initial setup.	Yes
35	I want to use the software on macOS computer.	Yes
36	I want to use the software on Windows computer.	Yes
37	I want to use the software on Linux computer.	Yes
38	I want a user manual for the software.	No
39	I want the implementation of analysis steps formally documented.	No

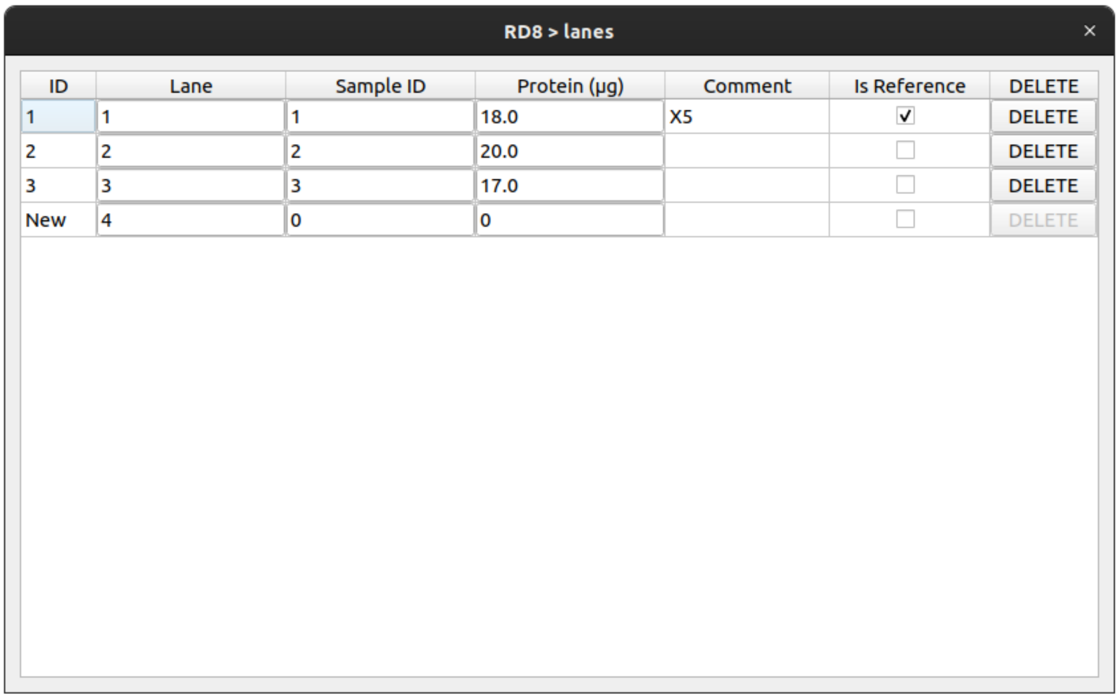
Table 6. Requirements of a scientist

Appendix 3 – Application Views



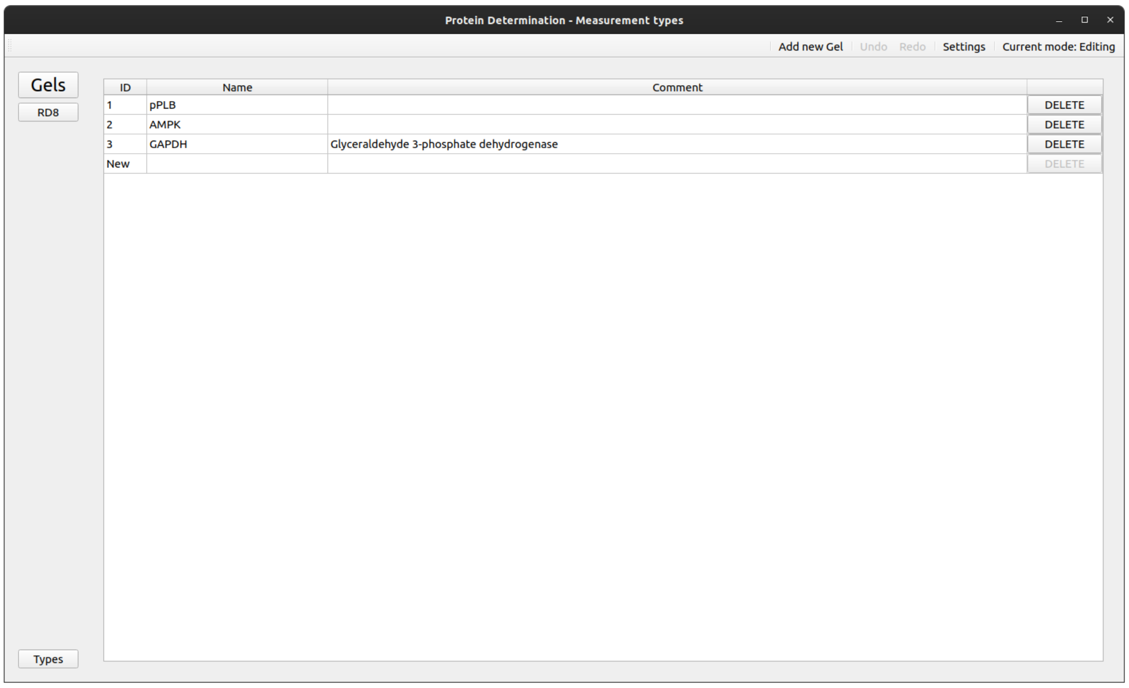
ID	Name	Transfer	Comment	Lanes	Measurements	
1	RD8	09.05.22 03:1	From Alex's lab	3	1	DELETE
New		09.05.22 14:5		0	0	DELETE

Figure 14. *Application gel list view.*



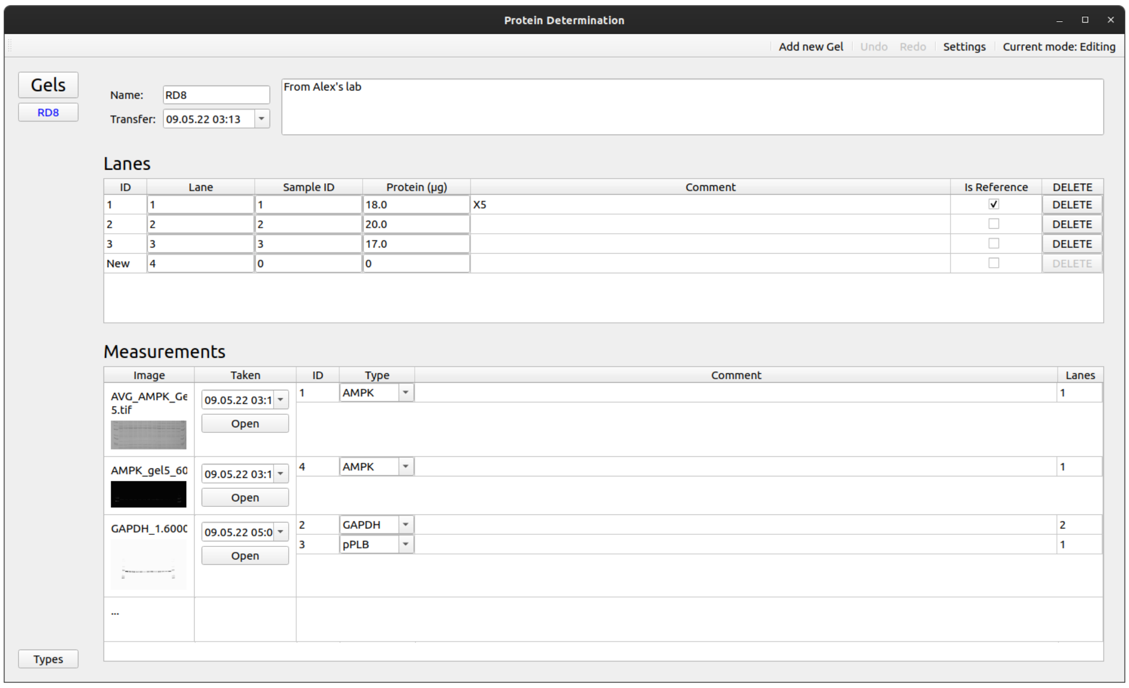
ID	Lane	Sample ID	Protein (μg)	Comment	Is Reference	DELETE
1	1	1	18.0	X5	✓	DELETE
2	2	2	20.0		☐	DELETE
3	3	3	17.0		☐	DELETE
New	4	0	0		☐	DELETE

Figure 15. *Application gel lanes popup view.*



ID	Name	Comment	
1	pPLB		DELETE
2	AMPK		DELETE
3	GAPDH	Glyceraldehyde 3-phosphate dehydrogenase	DELETE
New			DELETE

Figure 16. *Application measurement types list view.*



Name: RD8

Transfer: 09.05.22 03:13

From Alex's lab

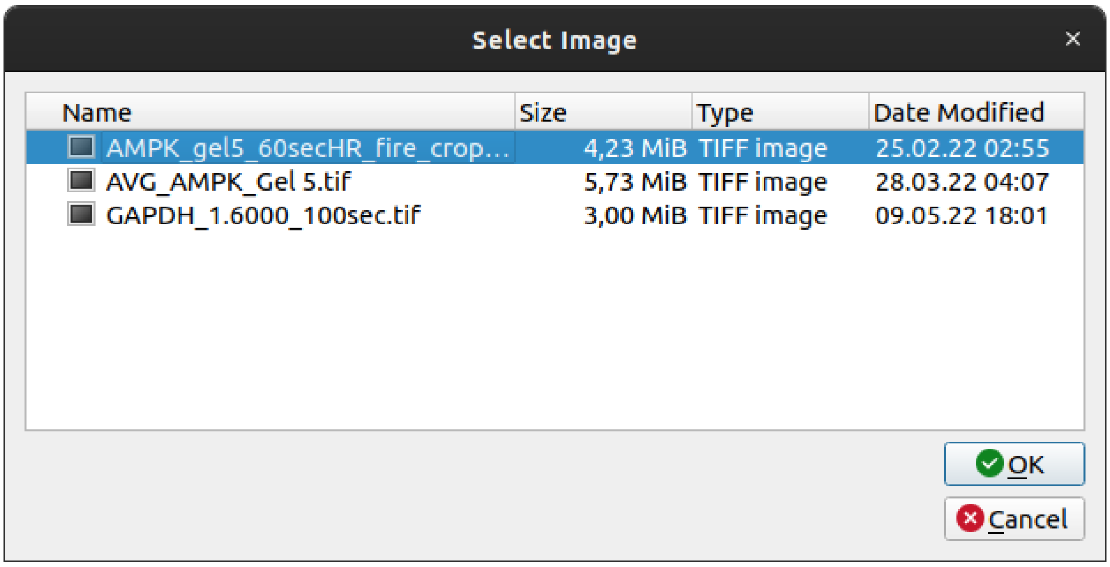
Lanes

ID	Lane	Sample ID	Protein (µg)	Comment	Is Reference	DELETE
1	1	1	18.0	X5	✓	DELETE
2	2	2	20.0		☐	DELETE
3	3	3	17.0		☐	DELETE
New	4	0	0		☐	DELETE

Measurements

Image	Taken	ID	Type	Comment	Lanes
AVG_AMPK_Ge 5.tif	09.05.22 03:1 Open	1	AMPK		1
AMPK_gel5_60	09.05.22 03:1 Open	4	AMPK		1
GAPDH_1.6000	09.05.22 05:0 Open	2	GAPDH		2
		3	pPLB		1
...					

Figure 17. *Application gel detail view.*



Name	Size	Type	Date Modified
AMPK_gel5_60secHR_fire_crop...	4,23 MiB	TIFF image	25.02.22 02:55
AVG_AMPK_Gel 5.tif	5,73 MiB	TIFF image	28.03.22 04:07
GAPDH_1.6000_100sec.tif	3,00 MiB	TIFF image	09.05.22 18:01

Figure 18. *Application image selection popup view (local folder).*

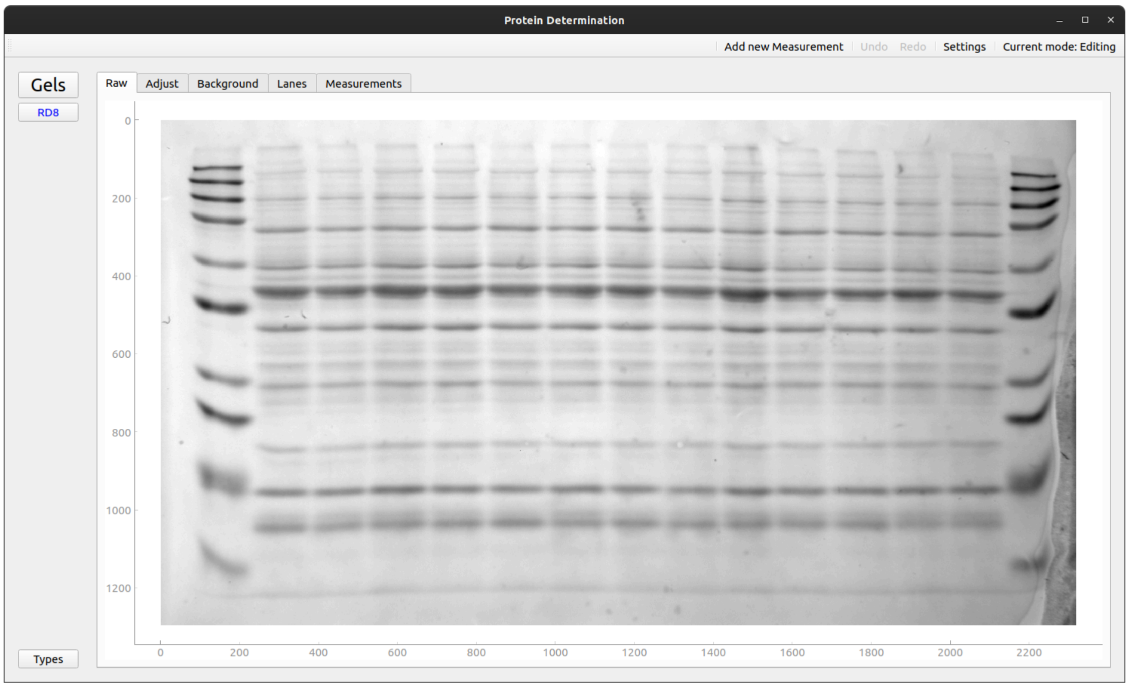


Figure 19. *Application image raw view.*

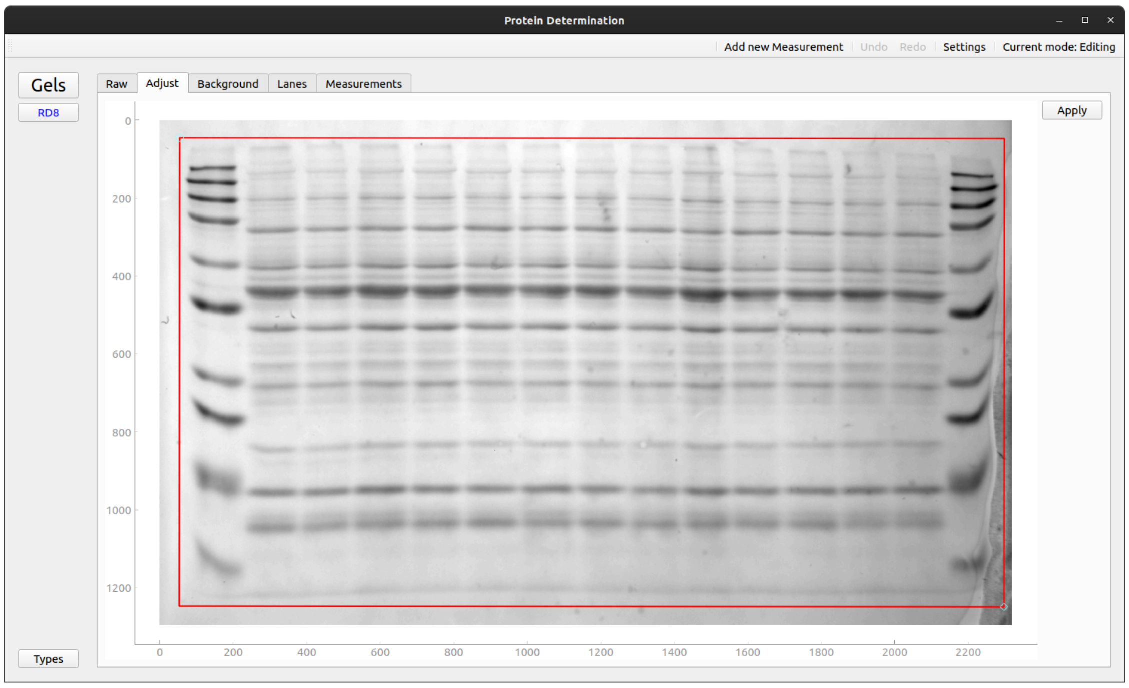
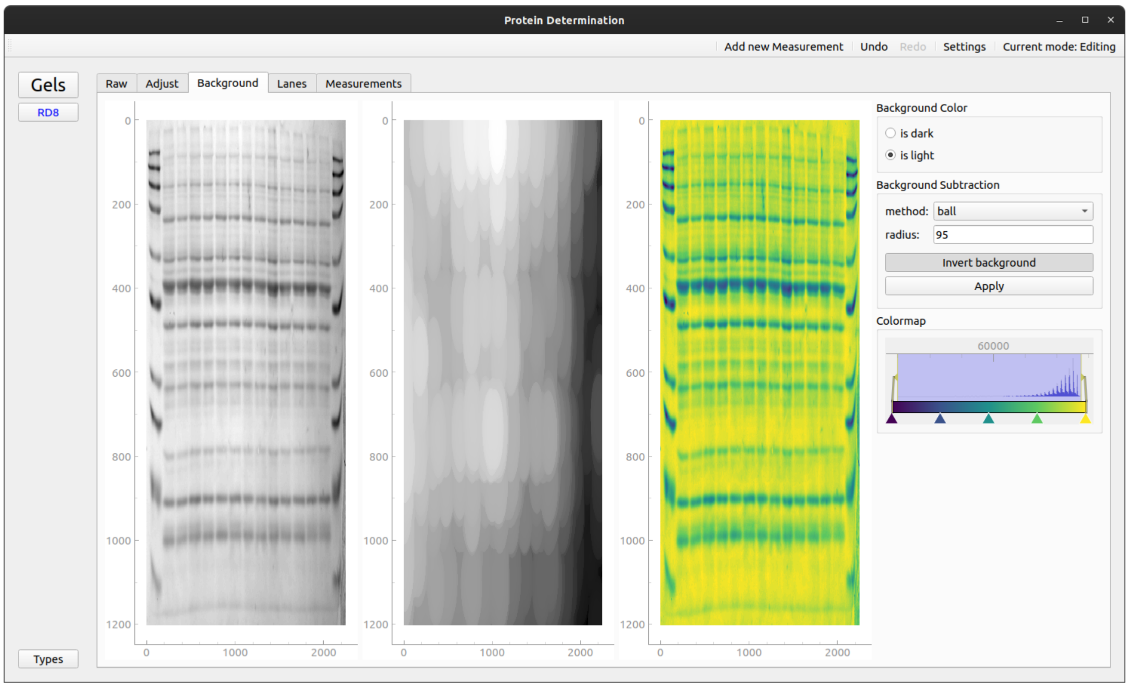


Figure 20. *Application image adjust view.*



Background Color

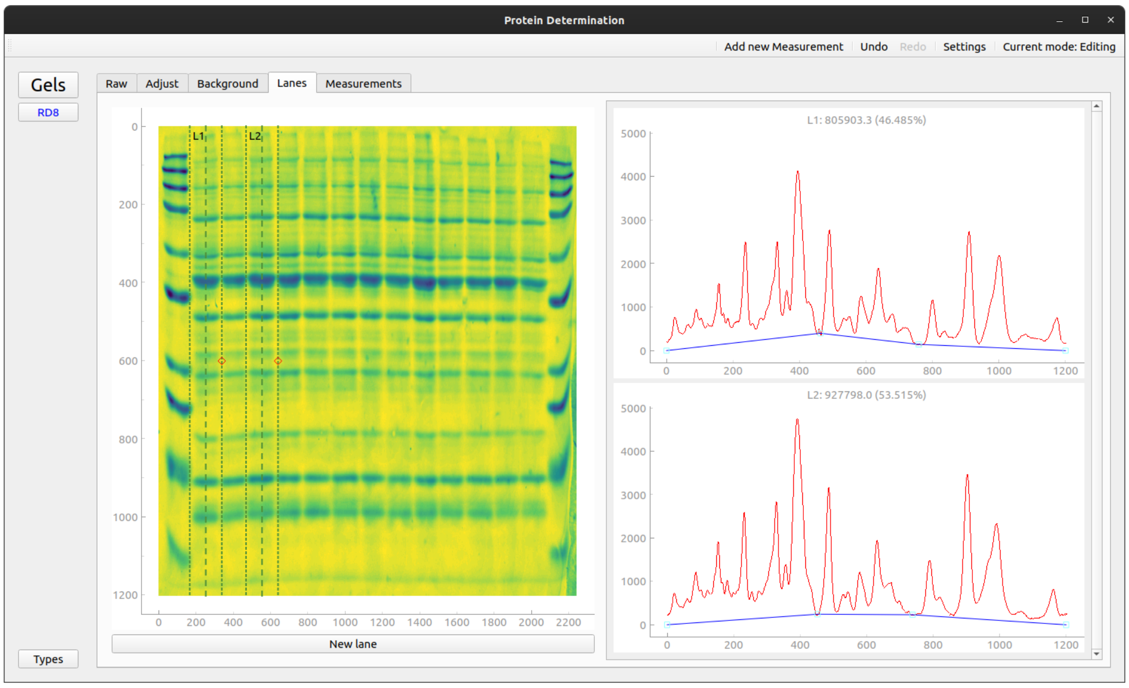
is dark [✓ is light]

Background Subtraction

method: ball

radius: 95

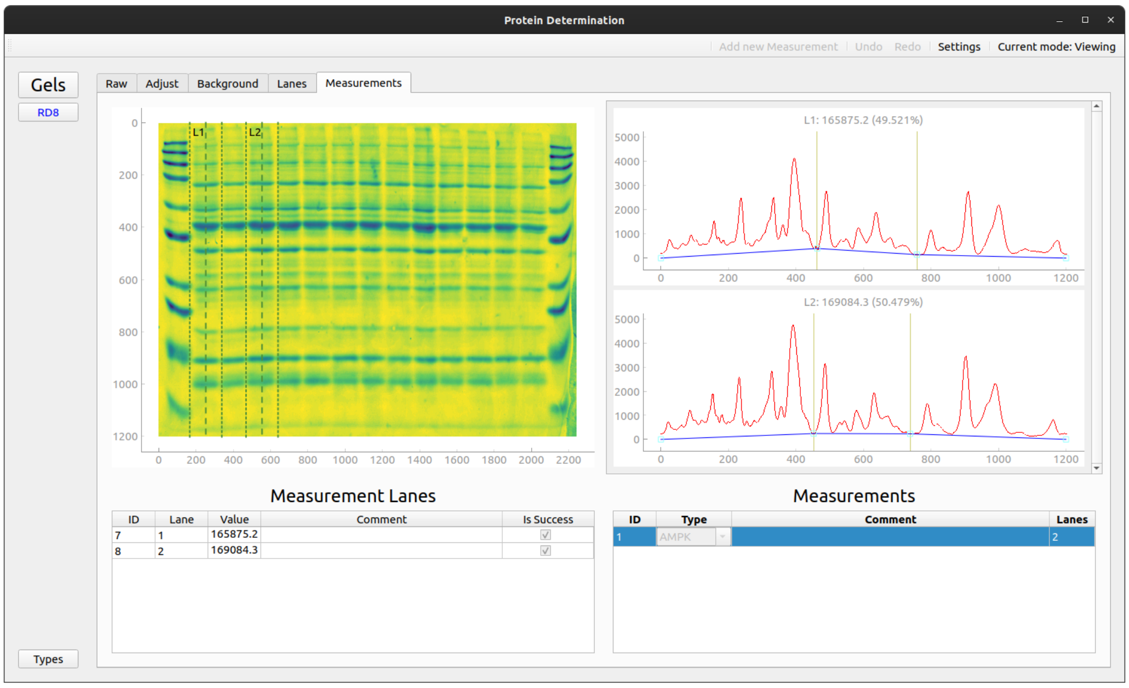
Figure 21. *Application image background subtraction view.*



L1: 805903.3 (46.485%)

L2: 927798.0 (53.515%)

Figure 22. *Application image lanes view.*



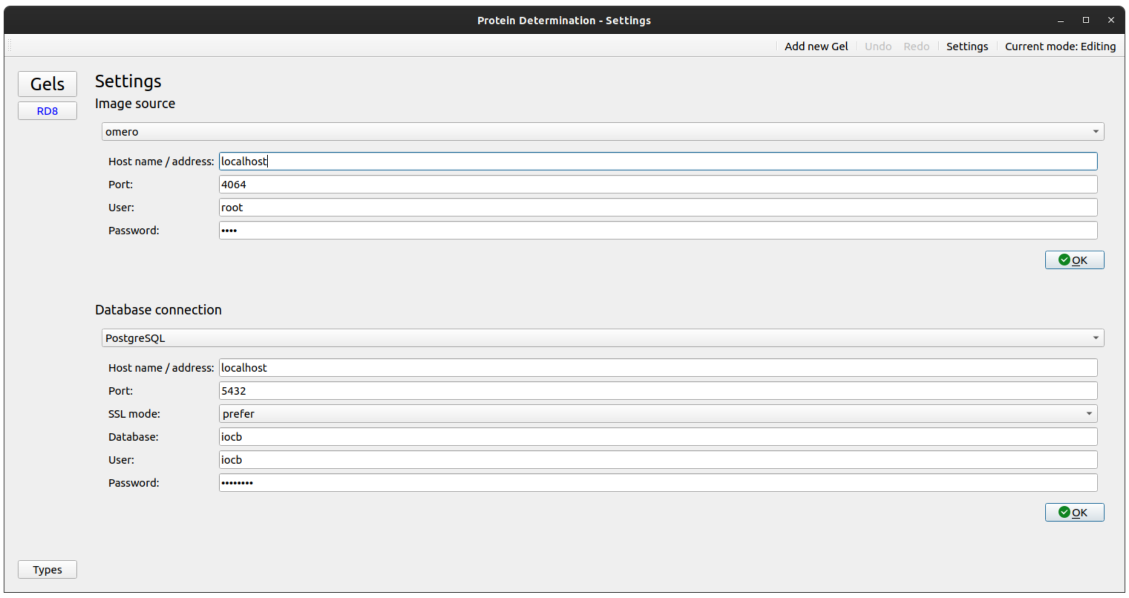
Measurement Lanes

ID	Lane	Value	Comment	Is Success
7	1	165875.2		✓
8	2	169084.3		✓

Measurements

ID	Type	Comment	Lanes
1	AMPK		2

Figure 23. *Application image measurements view.*
a) lanes graph, b) intensity plots
c) measurement lanes table, d) measurements table



Settings

Image source

omero

Host name / address:	localhost
Port:	4064
User:	root
Password:	••••

Database connection

PostgreSQL

Host name / address:	localhost
Port:	5432
SSL mode:	prefer
Database:	iocb
User:	iocb
Password:	••••••••

Figure 24. *Application settings view.*

Appendix 4 – Requirements Comparison

Table 7. Requirements satisfied by alternative applications

Requirement No.	**Requirement**	**ImageJ**	**Image Lab**
1	I want the software to be released under GPLv3 license.	Yes	No
2	I want GitLab pipeline configured to build packaged executables.	N/A	N/A
3	I want the software to look as agreed on the UI design document.	N/A	N/A
4	I want to import images from the lab's OMERO server.	No	No
5	I want my data saved on the lab's PostgreSQL server.	No	No
6	I want the software to indicate when it has lost external connections and allow to retry or change connection parameters.	N/A	N/A
7	I want to import images from my local computer.	Yes	Yes
8	I want my data to persist between work sessions locally.	No	No
9	I want to export the results of my work as a CSV file.	Yes	Yes
10	I want to use the software in my own language.	No	No
11	I want to add and modify measurement types.	No	No
12	I want to add and modify gels.	No	No
13	I want to add and modify the general information about the lanes on the gel.	No	Yes
14	I want to add multiple images for gels.	Yes	Yes
15	I want to view the raw image before analysis.	Yes	Yes
16	I want to select a ROI on the image before analysis (rotate and crop).	Yes	Yes
17	I want to specify if the background on the image is dark or light.	Yes	Yes
18	I want to correct image background (subtraction by rolling ball method).	Yes	Yes
19	I want to choose a false-color mapping for the image.	Yes	Yes
20	I want to mark the lane positions and widths on the gel image.	Yes	Yes
21	I want to adjust lane widths individually.	No	Yes
22	I want to rotate the lanes individually.	No	Yes
23	I want to curve the lanes individually.	No	Yes

Continues on next page

Table 7 – *Continued from previous page*

Requirement No.	**Requirement**	**ImageJ**	**Image Lab**
24	I want to see pixel intensity plots for marked lanes.	Yes	Yes
25	I want to specify a zero-line to separate background from signal on the intensity plots.	Yes	Yes
26	I want to add and modify measurements connected to specific gel images.	N/A	N/A
27	I want to choose which lanes a measurement includes.	Yes	Yes
28	I want to specify the integration limits on the pixel intensity plots.	Yes	Yes
29	I want the selected limits to be visible on their lanes.	Yes	Yes
30	I want to see a comparison and annotate analyzed lanes for measurements.	Yes	Yes
31	I want to search previously analyzed gels by name, ID, sample ID, lane ID and measurement ID.	Yes	Yes
32	I want the option to undo an accidental modification.	No	Yes
33	I want a way to disable all editing operations to avoid accidental modifications during a review.	Yes	Yes
34	I want to change my image and data source parameters after initial setup.	No	No
35	I want to use the software on macOS computer.	Yes	Yes
36	I want to use the software on Windows computer.	Yes	Yes
37	I want to use the software on Linux computer.	Yes	No
38	I want a user manual for the software.	Yes	Yes
39	I want the implementation of analysis steps formally documented.	Yes	Yes

Appendix 5 – Questionnaire Results

Table 8. Questionnaire: user roles

Which of the following roles apply to you?	**Answers**
I want to use the software only while working at the Systems Biology lab.	3
I am looking to use the software outside of the Systems Biology lab.	3

Table 9. Questionnaire: user confidence

Please answer how confident do you feel the following features work and meet your expectations on the scale of 1 (Not at all confident) to 5 (Very confident).	**1**	**2**	**3**	**4**	**5**
I can import images from the lab's OMERO server.	0	0	0	0	3
I can import images from my local computer.	0	0	0	0	3
My data is saved on the lab's PostgreSQL server.	0	0	0	0	3
My data persist locally between work sessions.	0	0	0	0	3
I can add and modify measurement types.	0	0	0	0	3
I can add and modify gels.	0	0	0	0	3
I can add and modify the general information about the lanes on the gel.	0	0	0	0	3
I can add multiple images for gels.	0	0	0	0	3
I can view the raw image before analysis.	0	0	0	0	3
I can select and rotate a region on the image before analysis.	0	0	0	0	3
I can specify if the background on the image is dark or light.	0	0	0	0	3
I can correct image background.	0	0	1	0	2
I can choose a false-color mapping for the image.	0	0	1	0	2
I can mark the lane positions and widths on the gel image.	0	0	0	1	2
I can adjust lane widths individually.	0	0	0	0	3
I can see pixel intensity plots for marked lanes.	0	0	0	1	2
I can specify a zero-line to separate background from signal on the intensity plots.	0	0	0	1	2
I can add and modify measurements connected to specific gel images.	0	0	0	0	3
I can choose which lanes a measurement includes.	0	0	1	0	2
I can specify the integration limits on the pixel intensity plots.	0	0	0	1	2

Continues on next page

Table 9 – *Continued from previous page*

Please answer how confident do you feel the following features work and meet your expectations on the scale of 1 (Not at all confident) to 5 (Very confident).	**1**	**2**	**3**	**4**	**5**
I can see a comparison and annotate analyzed lanes for measurements.	0	0	0	0	3
I can undo an accidental modification.	0	0	0	0	3
I can disable all editing operations to avoid accidental modifications during a review.	0	1	0	0	2
I can change my image and data source parameters after initial setup.	0	0	0	1	2
I can use the software on macOS computer.	1	0	2	0	0
I can use the software on Windows computer.	1	0	1	0	1
I can use the software on Linux computer.	0	0	0	0	3

Appendix 6 – Database Schema

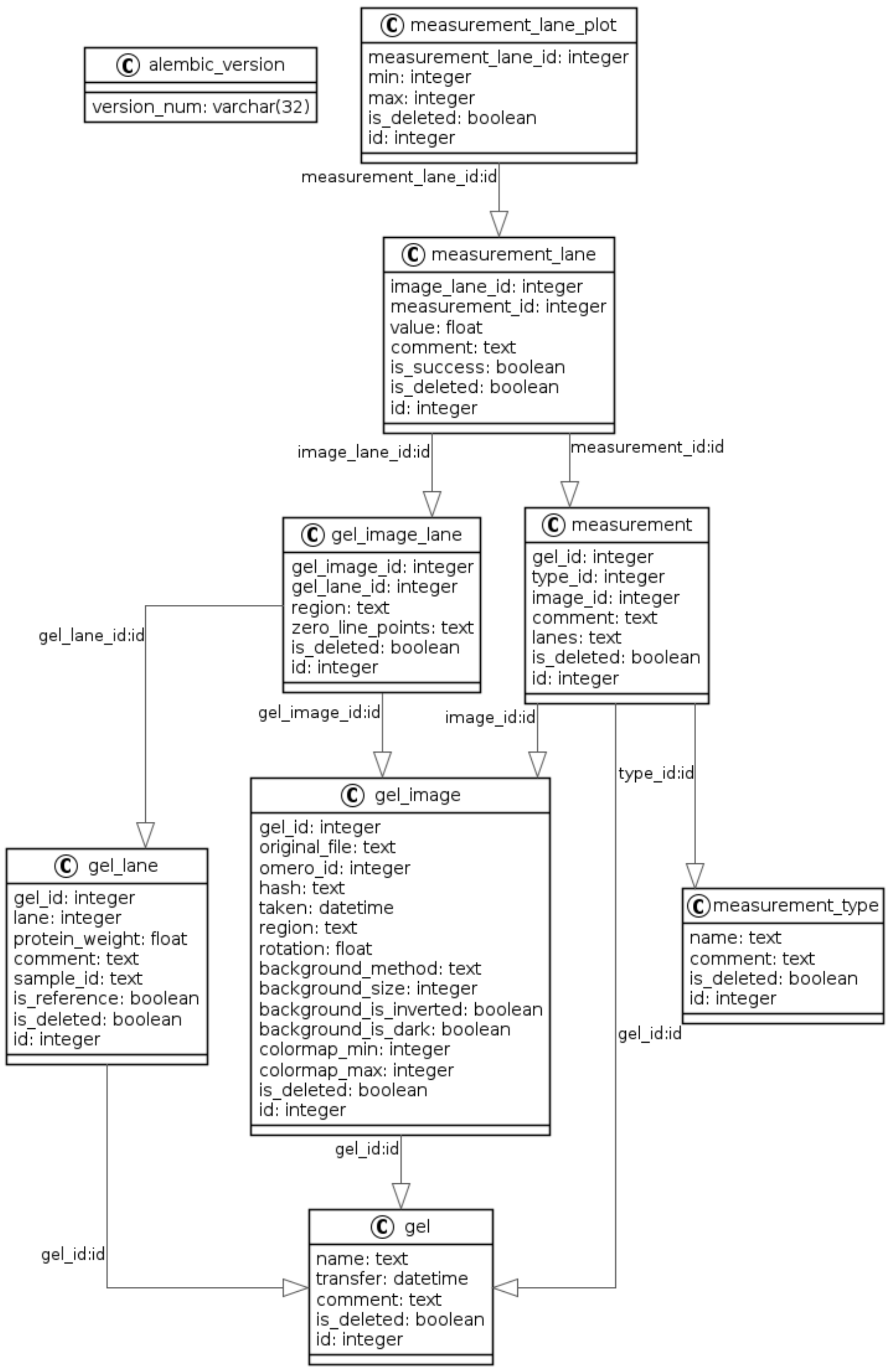


Figure 25. *Application database entity relationship diagram.*

Appendix 7 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

We Jaak Kütt, Georg Margus, Lauri Kask

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for our thesis "Software for Protein Determination", supervised by Priit Järv
    1.1. to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;
    1.2. to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
2. We are aware that the authors also retains the rights specified in clause 1 of the non-exclusive licence.
3. We confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

May 30th, 2022

[1]The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.