UNIVERSITY OF Southampton

Group 23: Deliverable 5: Group Report

COMP2211: Software Engineering Group Project

Jury D'Alessio (jd3n18), Mikolaj Kolybko (mk2u19), Rowan Kettle (rgk1g19), Velimir Nikolaev Anastasov (vna1u19), Charles Powell (cp6g18), Amir Abbasgholi Ghafghazi (aag1u18)

Electronics and Computer Science
University of Southampton

Contents

Introduction

This document details Group 23's evaluation of the product produced for COMP2211: Software Engineering Group Project. This report evaluates the Group's team work and time expenditure, considers the software tools used and their effectiveness, discusses the methods/strategies of communication, and provides advice to students who will undertake this module in the future. Contained in Appendix A is a URL that links to a demonstration video of the final product.

1 Evaluation of Team Work

1.1 Successes

The below list details the successes of the Group with regards to team work during project development.

- **Agile Methodologies**: Throughout the project duration, the Group's use of Agile Methodologies allowed for a more efficient development process, and a a higher quality application to be produced. Scrum development was used to implement the solution, with comprehensive Sprint planning taking place prior to each increment. Following the completion of a sprint, a sprint review allowed for Group members to assess the performance of the sprint, while a sprint retrospective gave the customer the opportunity to provide feedback on the developed product. The advantages and disadvantages of the Group's use of Agile Methodologies are considered in Review of Agile Methodologies.
- **Extreme Programming**: The Group made use of Extreme Programming with great success, resulting in a methodical and efficient product development. The Group took time at the start of the project to understand the five principles of Extreme Programming and the benefits it can provide, practicing these ideas from the start of the project and maintaining them throughout. The exact details of the Group's approach to Extreme Programming can be found within Review of Extreme Programming.
- **Planning**: Planning has been crucial part of the Group's project, and has been to used great effect. Planning took place prior to each increment and involved the Group deciding the tasks that needed to be completed in the upcoming sprint, distributing the tasks amongst the Group members, and organizing the tasks across the course of the increment timeline. The Group also used different planning strategies for each of the three sprints, intentionally placing a larger portion of the workload earlier on, as to avoid issues regarding commitments to other modules that may arise at later points in the project.
- **Design**: The Group carried out extensive design work at the start of each increment, making use of a wide range of UML techniques to ensure an efficient development. Placing such a large focus on design allowed for a structured and methodical increment to take place, and one in which the customer could gain an understanding of the planned product before development began, providing feedback where necessary.
- **Responding to feedback**: Following each sprint, the Group met with the customer and an additional examiner to demonstrate the product and receive formal feedback. Following these review meetings, the Group placed great focus on responding to the provided feedback, and making changes to the product or development process where required. This approach ensured that the application was valuable to the customer, meeting all of their requirements. Full details of the Group's response to feedback can be found within the Response to Feedback and Product Adjustment sections of the individual sprint reports.
- **Improvements with each increment**: At the start of each sprint, the Group took time to make changes to the product outside of the scope of it's requirements in order to increase it's usability and overall quality. An example of such changes include the addition of confirmation windows to the application to ensure that the user is not able un-intentionally perform actions. Full details of these adjustments can be found within the Product Adjustments section of the individual sprint reports. Making these adjustments and planning them around the scheduled product backlog items for the sprint ensured that the customer is provided with a product that meets their requirements, and offers them a pleasurable user experience.
- **Group support**: The Group consistently demonstrated an effective team relationship when supporting each other through challenges. The Group contained a diverse range of skill sets, and individuals accommodated for this by supporting those who were struggling in areas in which they were experienced. Examples of these challenges range from technical issues such as programming to language difficulties that present themselves when writing reports.

1.2 Challenges

The below list details the challenges the Group faced with regards to team work during project development.

- **Communication**: The Group suffered communication issues at times, particularly during the storming phase of group development. These issues lead to a lack of clarity in terms of the overall goal/aim of the product, and therefore hindered it's development process. These issues were overcome with time, as the group relationship developed, and the quality of communication improved.

- **Use of Git**: The Group faced issues using the Git tool collaboratively at the start of project development, as a result of the inexperience of some group members. These issues resulted in minor set backs to the development process, but were overcome by more experienced Group members offering their guidance and support.
- **Task distribution**: Task distribution has been a cause for concern throughout the majority of the project, with some group members taking on a larger share of the workload. While poor task distribution can lead to individuals becoming overwhelmed, and thus not completing the work to a high standard, the Group does not feel that the challenges faced have negatively impacted the quality of the final product.

1.3 Review of Agile Methodologies

1.3.1 Advantages

The following list details the benefits the Group experienced as a result of practicing Agile Methodologies.

- **Productivity**: The focus placed on planning, reviewing, and receiving feedback by Agile methodologies resulted in an efficient and productive development.
- **Flexibility**: Agile Methodologies increased the flexibility of the development process. While a focus was placed on planning before each sprint, Agile principles allowed for adjustments to be made to the plan following unforeseen circumstances. An example of this includes the moving of product backlog items to a future sprint after failing to implement them in the current sprint due to time constraints.
- **Risk of missing goals**: The flexibility and productivity provided by Agile Methodologies reduced the risk of missed goals and ensured that the customer received a product that met their requirement.
- **Customer involvement**: Agile Methodologies supported a greater customer involvement during development. The Group met once weekly with the customer during sprints to provide a progress report, and following the completion of a sprint to perform a product demonstration and receive formal feedback. Meeting frequently with the customer during development allowed for greater transparency, making it easier for the customer to voice their opinions and ensure that the application being produced met their requirements. Ultimately, this provided the customer with a more valuable product, and therefore the development can be seen as more successful.

1.3.2 Disadvantages

The following list details the challenges presented to Group as a result of practicing Agile Methodologies.

- **Project focus**: The Agile development structure can easily lead to the project losing focus and drifting away from the customer's requirements. This was an issue that the Group encountered during the first sprint, but were able to overcome for the remainder of the project through improved communication and thorough planning.
- **Predictability**: Following Agile Methodologies lead to a decrease in predictability during development. The structure of Agile development resulted in uncertainty about future events, making it difficult for the Group to plan ahead beyond the current sprint.
- **Time and commitment**: Following Agile Methodologies involved frequent meetings within the Group and with the customer to ensure an efficient development process that follows the customer's requirements. However, such an approach to development increases the commitment required from the Group members, which can become burdensome when coupled with the workload already faced.

1.4 Review of Extreme Programming

The Group's approach to Extreme Programming is detailed below in terms of the five key principles, with descriptions on how these principles were followed and the benefits they provided to the development process.

1.4.1 Communication

Communication has been a key focus of the Group from the start of the project. The Group made use of Discord and Microsoft Teams to maintain a strong line of communication between each other, and with the supervisor respectively. Regular meetings were held as a group, and weekly with the supervisor, ensuring that every member of the Group is able to contribute their ideas to the development process, and that issues are handled as a collective.

Placing such a focus on communication within the project has ensured that development takes place in the most efficient way possible, and that the final product is of the highest possible quality.

1.4.2 Simplicity

Simplicity aims to avoid wasting time by implementing only what is absolutely necessary and ignoring extensions until this is complete. Throughout this project, the Group has striven to practice these ideals to ensure that the final product is guaranteed to meet the customer's requirements, and is therefore of value to them. Such simplicity is evident in the application GUI, where the Group chose a minimalist design, focusing on functionality over appearance. This was a choice that pleased the customer, who noted that their liking of the application's "simplistic" and "intuitive" design. The Group's use of prioritisation also demonstrated simplicity in that extensions were only considered in the final sprint of development, after all of the customer's initial requirements were implemented.

1.4.3 Feedback

Feedback has played a crucial part in the project development. During development sprints, the Group met once weekly with the Supervisor to showcase recent work and seek informal feedback. The Group would then made slight adjustments to the product before the end of the sprint according to this feedback. After each sprint, a review meeting would take place, in which the product would be demonstrated, and the customer would provide formal feedback alongside a secondary examiner. After receiving this feedback, the Group would take time at the start of the following sprint to adjust and improve the product, before beginning development on further product backlog items. This diligent approach to feedback has ensured that the final product meets all of the customer's requirements, and is therefore a valuable application.

Full details of the adjustments made to the product in each sprint, along with the Group's response to other feedback are contained within the Product adjustments and Response to Feedback sections of the respective sprint report.

1.4.4 Courage

Group members have displayed great courage when voicing their concerns during development to encourage changes to be made. A notable example of this occurred in the first sprint, where a Group member took time to raise issues relating to the fundamental design of the application, in fear that it's current state would not support the full extent of the customer's requirements. As a result of this, the Group took the time to re-structure the application, producing a higher quality product as a result.

The Group's willingness to take on board feedback and adjust the product to better suit the customer's requirements also demonstrates courage. Before making the necessary adjustments, the Group first had to accept that there were aspects of the product that the customer was not pleased with, a task that can be challenging when considering the amount of effort placed into it's development.

1.4.5 Respect

The Group practiced respect by maintaining a healthy and constructive relationship throughout the course of the project. This involved aspects such as ensuring that all Group members were able to provide their opinions on development decisions, creating an environment that allows for issues to be raised by Group members, and supporting Group members during challenging times. In a Group project such as this, maintaining a respectful relationship within the development team is a vital part of ensuring an efficient and effective development process.

This level of respect was extended to the supervisor at all times. During meetings or general discussions, the Group would respect the authority of the supervisor, and appreciate any and all feedback provided by them. As the supervisor has the role of customer within the project, developing a respectful relationship with them ensures that the Group and supervisor are able to effectively work together, and therefore that the customer receives the product they ultimately desire.

2 Time Expenditure

2.1 Sprint Analysis

The below table provides an outline for each of the three development sprints and considers their respective time expenditure. Note that the Group allocated the most time to Sprint 1, followed by Sprint 2 followed by Sprint 3. This was done with intention, as the Group aimed to complete as much of the work as possible at the start of the module, where Group members have the least commitment to other modules (i.e., no other coursework to complete).

Contained in Appendices B, C and D are product screenshots, sprint burn-down charts and links to the full sprint reports for sprints 1, 2 and 3 respectively.

	Sprint 1	Sprint 2	Sprint 3
Estimated Workload	180 Hours	120 Hours	80 Hours
Rundown	• Developed core system functionality	• Adjusted product according to customer feedback • Extended application to allow for the graphical display of runway revisions • Implemented notification system	• Adjusted product according to customer feedback • Extended application to allow importing/exporting of airports/obstacles • Extended application to allow exporting of runway revisions • Extended application to allow exporting of system notifications • Created User Guide
Testing	• JUnit (backend) testing	• JUnit (backend) testing • JavaFX testing • Scenario testing	• Scenario testing
Expensive Tasks	• Developing application GUI • Developing main system operations	• Graphical display of runway revisions	• Bug fixing • Importing functionality
Successes	• Effective time management • Sticking to sprint plan • Extensive product design	• Excellent Response to Feedback • Efficient Development Process • Extensive Testing	• Workload Distribution • Improvements • Manual Testing • Agile Methodologies
Challenges	• Version Control issues • Defining system scope • Distribution of tasks	• Workload Management • Identifying most suitable graphics package	• Time constraints • High quality user guide

2.2 Overall Analysis

2.2.1 Workload Estimation

During the planning stage of each sprint, effort estimation was provided via Story Points using the Fibonacci Series as the story point range. These estimations were made using the online tool Planning Poker, which allowed for an un-biased and collaborative decision to be made on the estimated workload of a given task. The Group would overestimate the required workload for each task, as to ensure that the increment is always completed before the end of the sprint.

After completing the workload estimation, tasks could be prioritised based on their respective estimated workload, and the sprint could be effectively planned to ensure an efficient development process.

2.2.2 Individual Contributions

The below table lists the estimated contribution of each member of the Group in terms of hours spent on the project per week.

Name	Estimated Contribution (Hours/Week)
Charles Powell (cp6g18)	21 - 23
Amir Ghafghazi (aag1u18)	11 - 12
Jury D'Alessio (jd3n18)	7 - 8
Mikolaj Kolybko (mk2u19)	11 - 12
Velimir Anastasov (vna1u19)	7 - 8
Rowan Kettle (rgk1g19)	7 - 8

2.2.3 Improvements

Listed below are the improvements that the Group would make in relation to the time expenditure of the project development.

- **Task distribution**: The Group agrees that a poor task distribution resulted in an uneven spread of workload across group members, as seen in the above table. Poor task distribution can often lead to certain group members becoming overwhelmed with their work, and thus not completing it to a high enough standard, while other group members are left with little to do. Whilst the Group does not feel this task distribution negatively impacted the quality of the finished project, they acknowledge the improvements that could have been made, and will take this knowledge forward into future projects.
- **Workload estimation**: The Group agrees that more focus should have been placed on workload estimation during the planning phase of each increment, as to allow for a better distribution of tasks. A more accurate workload estimation ensures that tasks can be better distributed amongst group members as each group member can be allocated tasks of the same estimated workload. Evenly spreading the workload ensures that the project is developed in the most efficient way possible, and thus that the final result is of the highest possible quality.

3 Software Tools

The below table lists all of the software tools used during the project, providing a definition and description of value for each. The effectiveness of each tool is also ranked according to a three point scale to provide an indication of the tool's importance (high, medium and low).

Name	Description	Value	Effectiveness
Trello	*An online based tool that provides project organisation by listing project tasks within a 'Board'. A 'Board' divides tasks into a number of sections based on their classification/status (e.g., to do, currently being done, and completed). User's can create and delete sections, or move tasks between them as their corresponding status changes.*	Used to organize project tasks during each development sprint. A new 'Board' was created for each stage of the project, which contained all of the tasks to be completed during the increment. Each task could then be assigned to a member of the group, and the status of the task could be tracked using the different sections of the board, and by adding comments to the task.	MEDIUM

Discord	*An instant messaging and digital distribution platform designed for creating communities. Users communicate with voice calls, video calls, text messaging, media and files in private chats or as part of communities called "servers".*	Served as the primary form of communication between the Group members. This included general discussion through text channels, and sprint meetings held through voice channels. Discord provided value through it's simple interface that allows for information to quickly and easily be exchanged in a variety of formats. As Discord also has a Mobile Application, Group members were able to stay in touch with each other, even when they did not have access to their primary computer.	**HIGH**
Microsoft Teams	*A communication platform used by organisations, including the University. In Microsoft Teams, a number of people can form a group, known as a 'Team', allowing them to exchange information through different channels, and engage in live conversations through 'Meetings'.*	Used for formal communication between the group members and the supervisor. This included general discussion as well as formal review meetings held after every deliverable. Using Microsoft Teams in this way allowed the Group to separate inter-group communication with that between the Group and the Supervisor.	**HIGH**
Overleaf	*A collaborative cloud-based LaTeX editor used for writing, editing and publishing documents.*	Used to create the necessary documentation for the project. Overleaf provides value as it allows for real time collaboration on documents, and presents them in an incredibly structured and formal manner.	**HIGH**
Google Drive	*A file storage and synchronization service developed by Google. Google Drive allows users to store files in the cloud, synchronize files across devices, and share files.*	Used on occasion to share documents that were otherwise too large to be shared through Microsoft Teams or Discord - for example, the compiled application as a '.jar' file.	**LOW**
Planning Poker	An online tool that provides a consensus-based, gamified technique for estimating the effort or relative size of development goals in software development.	Used during the planning stage of each increment to generate story point estimations for the product backlog items contained in the sprint. Planning Poker provides value as it considered the opinions of all group members to create an un-biased estimation of the effort required to complete a given task.	**LOW**
Visual Paradigm Online	*A cloud-base collaboration platform that allows for users to store and share Visual Paradigm projects.*	Used to generate UML diagrams during the design stage of each development increment. This included Class and Sequence Diagrams as well as User Scenarios. Visual Paradigm Online provides value as it presents structured diagrams, and allows for multiple Group members to collaborate on the diagrams at any given time.	**MEDIUM**
IntelliJ IDEA	*An integrated development environment that supports the development of computer software. IntelliJ IDEA is primarily used in the production of software being written in the Java programming language.*	The primary tool used for the purposes of code-base development and management. IntelliJ IDEA is optimised for the Java programming language and provides an abundant range of features to allow for an efficient development process, including support for Git, Maven and testing frameworks.	**HIGH**

Git & GitLab	*Git is a Version Control System (VCS) used for recording changes to sets of files. GitLab is a web-based Git-repository manager that allows for multiple individuals to collaborate on a single Git project.*	Git and GitLab were used in combination to record project changes, and to allow for project collaboration across the various machines of individual group members. The use of these tools ensured an efficient development process, and one in which any application errors could be swiftly resolved.	**HIGH**
Maven	*A build automation tool used primarily for Java projects, designed to take much of the hard work out of the building process.*	Used to organise the project dependencies and automate the build process. The complexity of the project resulted in a large number of additional dependencies being required, which were easily managed through the use of the Maven tool.	**HIGH**
JavaDoc	*A documentation generator tool, used for generating structured documentation of code written in the Java programming language.*	The JavaDoc system, and it's corresponding commenting format, were used to generate a structured documentation of the code-base. Such documentation ensures that any future developers are easily able to understand the existing code-base, providing maintainability and expandability.	**HIGH**
JUnit	*A unit testing framework for the Java programming language.*	Used to test program functionality throughout the development process. The use of JUnit ensured that the application was performing operations correctly, such as the calculation of runway revisions.	**HIGH**
TestFX	*A GUI testing framework for the JavaFX package of the Java programming language. The TestFX framework assesses the correctness of applications by taking control of the pointer and keyboard, completing a set of pre-defined tasks, and comparing the application state to the provided expected state.*	Used to test that the project application was both functioning correctly and responding to user input appropriately. TestFX allowed for automation of scenario tests, which in turn provided efficient regression testing during the various stages of project development.	**HIGH**

4 Communication

4.1 Inter-Group Communication

4.1.1 Medium

The **Discord** application was for communication between group members. A server was established at the start of the project containing all six group members, and a number of channels were created within the server to distinguish between the various discussions that could take place (e.g., general text-discussions and voice-meetings). Individual group members also used Discord's private messaging feature to communicate directly with each other, such as in the case of "buddy"-programming.

4.1.2 Strategy

The Group's communication strategy is summarised below, detailing how information was exchanged, and the frequency of meetings.

Information Exchange

- A **General** channel was used for general discussions between group members throughout the course of the project.
- A **Resources** channel was used to share project resources, such as links to collaborative documents, or code-base files.
- A **Git Notifications** channel was configured to display notifications when ever a group member made changes to the project's remote Git repository.

- A **Meeting Plans** channel was used to record meeting plans for both meetings within the Group, and between the Group and supervisor.
- A **Meeting** channel was used to host live-discussions between Group members over voice and video. Discord also supports the sharing of screens during meetings, which was used by the group to demonstrate aspects of the product during development. Meeting notes were recorded during all meetings, and a tool was used to record both the audio and video from the meeting.

Meeting Frequency

- A meeting took place at the **start of an increment**, that was used to discuss/plan necessary work for the upcoming deliverable.
- Meetings took place every 2-3 days **during an increment**. The topic of discussion of these meetings varied depending on their relative timing within the increment.
- A meeting took place at the **end of an increment** (i.e., penultimate day before increment deadline), which was used to finalise the work and ensure it was ready for hand-in.

4.2 Group-Supervisor Communication

4.2.1 Medium

The **Microsoft Teams** application was used for communication between the Group and the supervisor. A text channel was established within the Team for general discussion, while the Meeting functionality allowed for review meetings to take place between the Group and the supervisor.

4.2.2 Strategy

The Group's communication strategy with the supervisor is summarised below, detailing how information was exchanged, and the frequency of meetings.

Information Exchange

- A **General** channel was used for general discussions between group members and the supervisors. This included informal questions relating to the project, as well as the organisation of meetings.
- A **Files** channel was used to share project files with the supervisor, such as submission documents required when performing review meetings.
- The **Meeting** system was used to host voice and video meetings between the Group and the supervisor. Group members were able to make use of the 'Share' functionality to share their screen during a meeting, allowing for product demonstration. Meeting notes were recorded during all meetings, and a tool was used to record both the audio and video from the meeting.

Meeting Frequency

- A **progress report meeting** took place every week, which was used to discuss the recently completed work, and gain some informal feedback from the supervisor.
- A **review meeting** took place in the days after a submission, and was used by the group to conduct a formal demonstration of the completed work, and to gain formal feedback from the supervisor.

5 Advice to Future Students

Listed below is the advice that Group 23 would present to future students of COMP2211: Software Engineering Group Project. This advice reflects the experiences of the Group during the module, and is generalised as to be applicable to students undertaking either of the available projects.

- **Keep it simple**. Do not try to "overshoot" the project, as it is likely it will require a lot more work than you initially think. A simple but complete project is better than a complex but unfinished project. Keep your project goals simple, and consider extensions when the core system is finished.
- **Understand that you will always underestimate the workload of tasks**. As this will most likely be the largest project you have undertaken, it is important to understand the amount of work required to complete it to a high standard. A common cause for error is underestimating the workload of tasks, and therefore not leaving enough time to complete them to a high enough standard. It is always best to overestimate, as this ensures that the task will be completed before it's respective deadline.

- **Communicate frequently and effectively**. Communication is a vital part of any project, and more so in this one as it will be the largest project you have undertaken at University so far. Being able to effectively communicate project goals, reflect on progress and plan for upcoming tasks will greatly increase the quality of the final product.
- **Be disciplined with project meetings**. As mentioned above, your ability to communicate as a group will reflect the quality of your product, and your attitude towards project meetings is a key part of this. Discipline involves being punctual, structuring meetings around the availability of the entire team and ensuring that meetings are regular, focused and purposeful. Meetings with the supervisor should not be excluded from this - they are your only chance to demonstrate your work and receive feedback and so should be treated seriously.
- **Keep meeting notes**. The importance of meetings should not be underestimated, and meeting notes should be used to record the important details discussed during them. Additionally, software can be used to record the audio and video from meetings to provide a more substantial record.
- **Don't forget about the increment reports**. Increment reports should not be forgotten about as they are one of the only records of the work you have completed. It is easy to place too much focus on product development, and not leave enough time to write the report. The result of this is a poor report that does not reflect the quality of your product, and thus you will not receive the marks you deserve.
- **Make use of your supervisor**. Your supervisor will act as the product customer, providing you with feedback during your development, but is also there to support you. It is important that you see your supervisor not only as a marker, and make use of their knowledge. Your supervisor will completed this module already, and so will have a wide range of advice to share with you.
- **Respond to feedback**. You will be provided with formal feedback throughout the course of the development process, and it is vital that you take this feedback on board and adjust the product accordingly. This module aims to demonstrate the reality of software development, and customer feedback is a crucial part of this. Failure to respond to feedback will result in a product that does not meet the requirements of the customer, and thus has little value.
- **Get ahead at the start and stay ahead**. The structure of the semester you undertake this module in is such that you will have a large portion of free time at the start, and very little towards the end. Therefore, taking the time at the start of the semester to get ahead on the project will benefit you in the long run, and ensure that you do not miss out on product aspects due to a lack of time.
- **Prioritise planning**. Planning is a crucial aspect in any development process, and should not be underestimated here. A heavy focus on planning throughout the project will ensure a smooth development process, and a higher quality product overall. Planning involves aspects such as setting deadlines outside of those already included in the module, effectively breaking down the project into the respective increments and dividing tasks between group members based on their preferences/skill-sets.

A Product Demonstration

The following URL links to a video that provides a full demonstration of the final product.

`https://youtu.be/gBGcshrvSaw`

B Supplementary Sprint 1 Information

B.1 Sprint Report

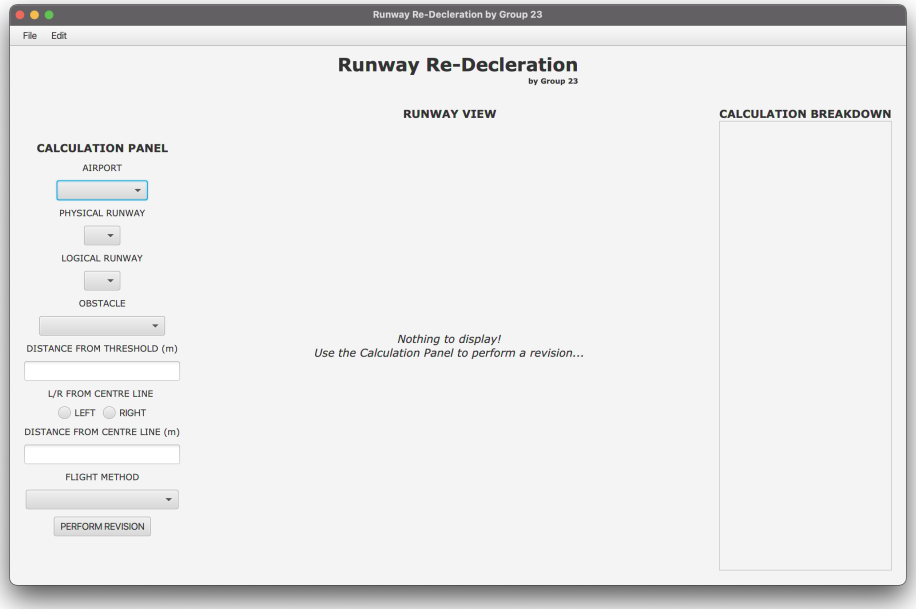
`https://bit.ly/2RJYzds`

B.2 Product Screenshots

Dashboard

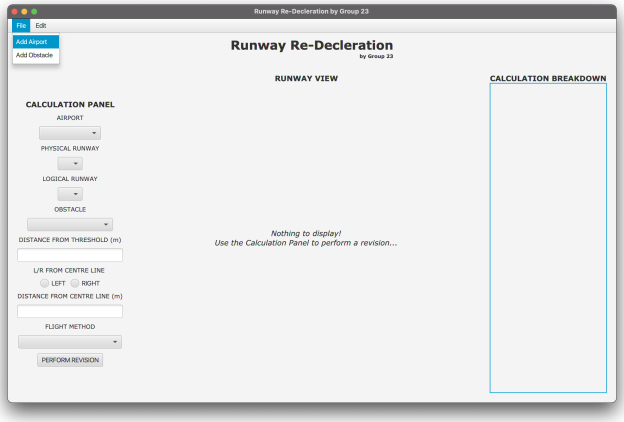
The dashboard is presented to the user when the application opens, and functions as the system home screen. From the dashboard, the user is able to perform runway revisions, review their results, and configure the persistent storage of the application (i.e., airports and obstacles).

The dashboard can be divided into a 'Toolbar', a 'Calculation Panel', 'Runway View' and 'Calculation Breakdown'. Shown below is a screenshot of the dashboard that the user is shown upon application startup.

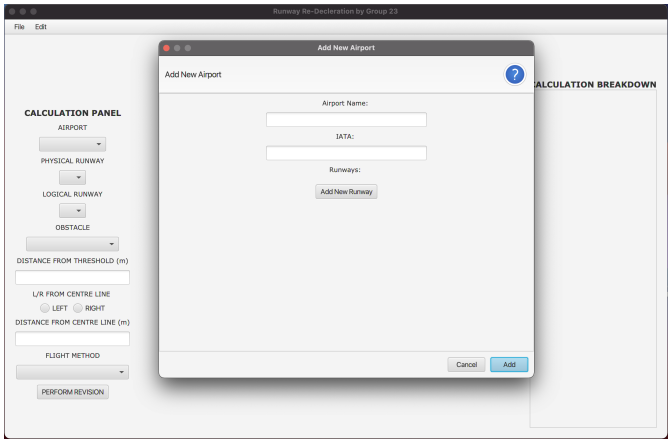


Declaring New Airports

The below screenshots shows the system allowing the user to declare a new airport. The user is able to open the form to configure a new airport by navigating to the 'File' menu within the toolbar. The persistence, of a newly created airport in memory can be seen when running the application.



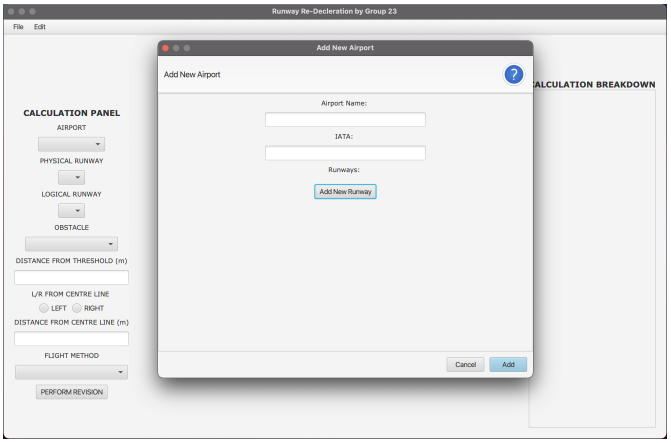
(a) Toolbar: 'File' Menu



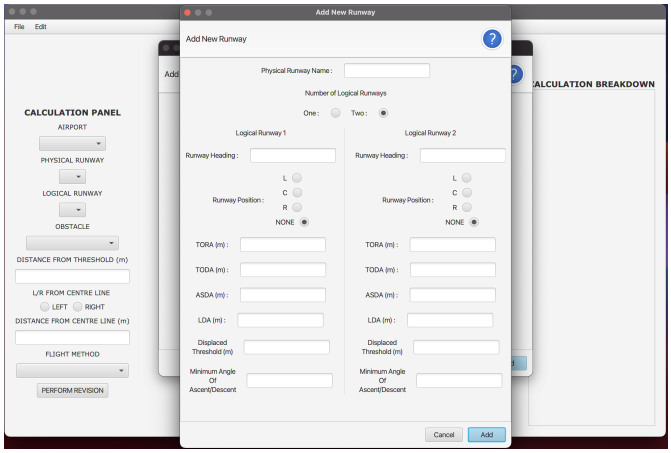
(b) 'Add New Airport' Form

Declaring New Runways

The below screenshots shows the system allowing the user to declare a new runways. The persistence, of a newly created runway in memory can be seen when running the application.



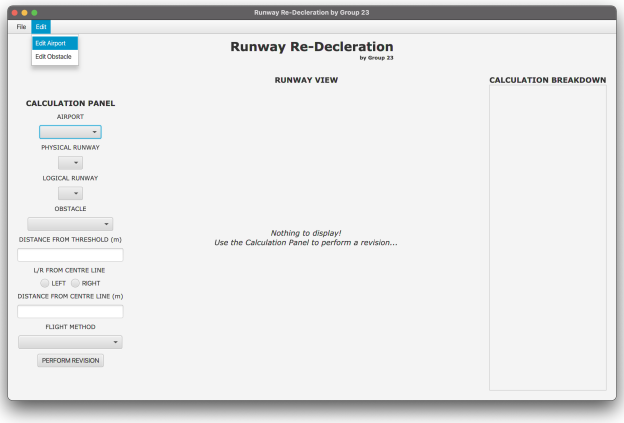
(a) 'Add New Airport' Form



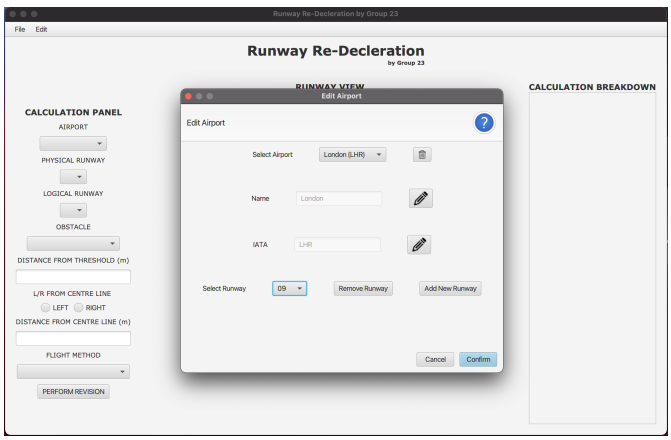
(b) 'Add New Runway' Form

Editing Existing Airports

The below screenshots shows the system allowing the user to edit the details of an existing airport. The user is able to open the form to configure an existing obstacle by navigating to the 'Edit' menu within the toolbar. From within the form, the user is able to select an existing airport, and can then configure the airport parameters, add/remove runways from the airport and remove the airport from the system entirely.



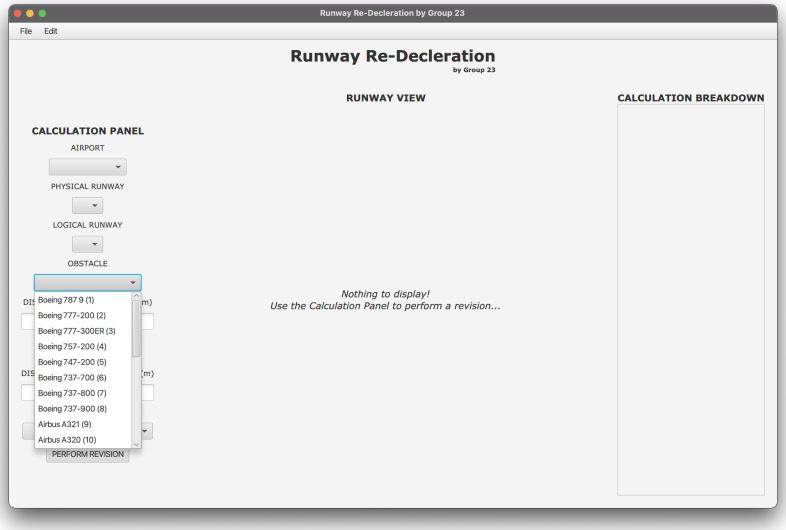
(a) Toolbar 'Edit' Menu



(b) 'Edit Obstacle' Form

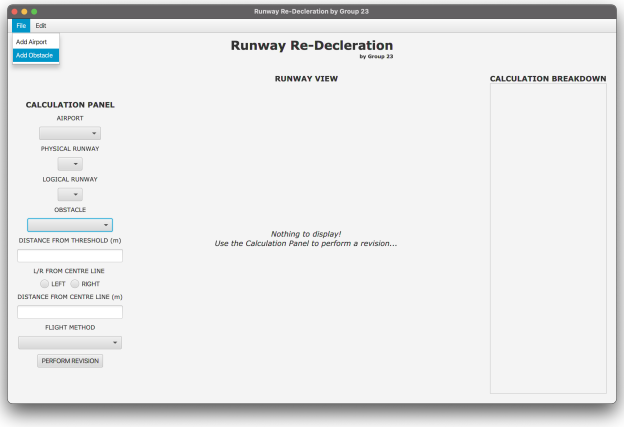
Predefined Obstacle List

The below screenshot show the some examples of the selection of obstacles that are predefined within the system, and ready for use in runway revision calculations.

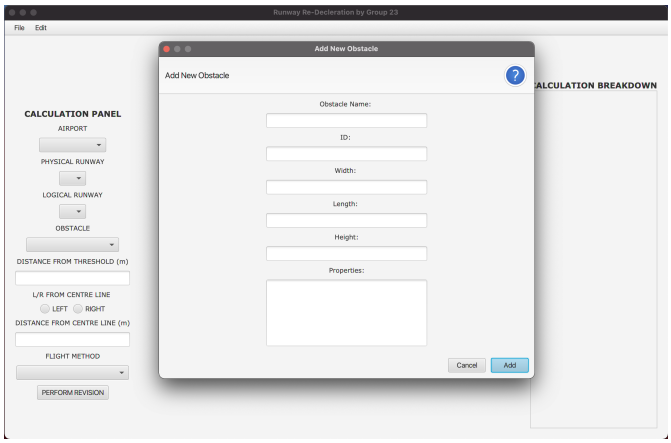


Declaring New Obstacles

The below screenshots shows the system allowing the user to declare a new obstacle. The user is able to open the form to configure a new obstacle by navigating to the 'File' menu within the toolbar. The persistence, of a newly created obstacle in memory can be seen when running the application.



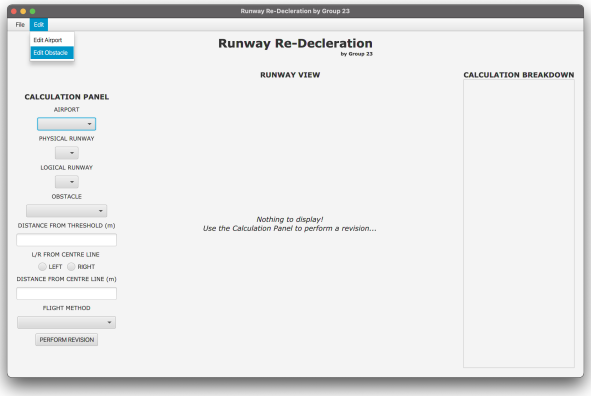
(a) Toolbar 'File' Menu



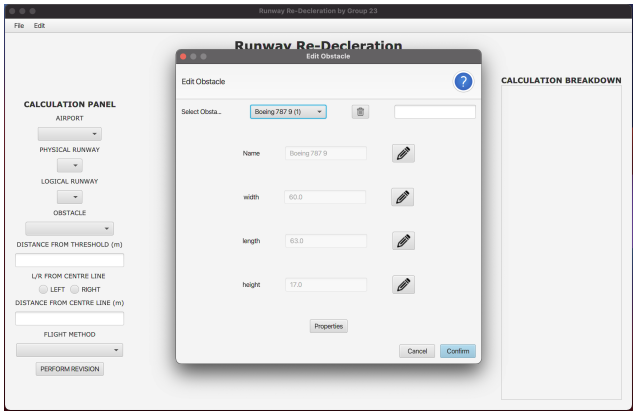
(b) 'Add New Obstacle' Form

Editing Existing Obstacles

The below screenshots show the system allowing the user to edit the details of an existing obstacle, runways. The user is able to open the form to configure an existing obstacle by navigating to the 'Edit' menu within the toolbar. From within the form, the user is able to select an existing obstacle, and can then configure the obstacle parameters, or remove the obstacle from the system entirely.



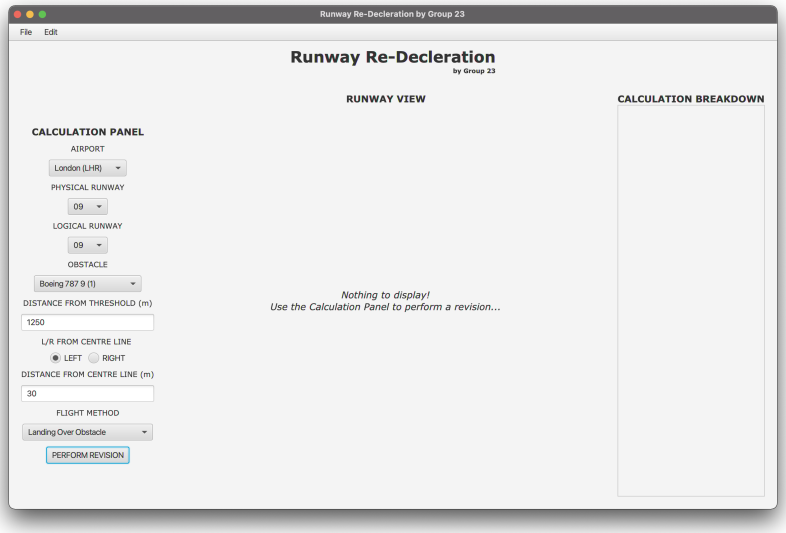
(a) Toolbar 'Edit' Menu



(b) 'Edit Obstacle' Form

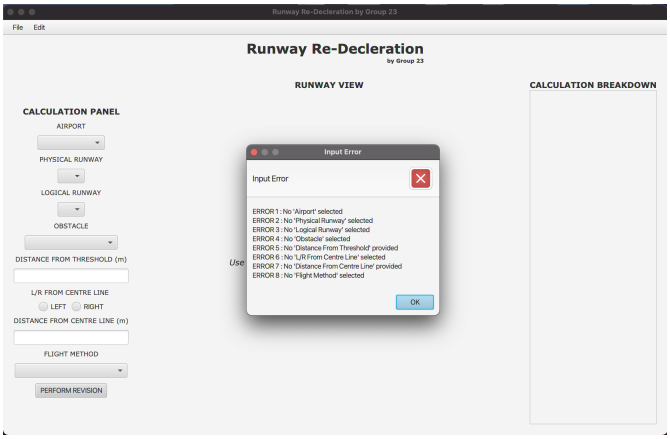
Positioning Obstacles for Runway Revisions

The below screenshot shows the system allowing the user to declare the placement of an obstacle within the 'Calculation Panel' of the dashboard. The user can then submit the runway for revision by selecting the 'Perform Revision' button.

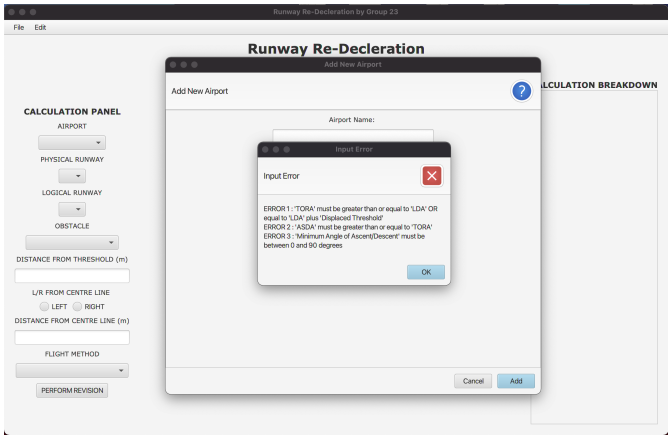


Input Error Checking

The system will validate all user input, ensuring that it is of the correct format, and that it follows all necessary constraints. The below screenshots show examples of error alert windows that are shown to the user when invalid information is submitted.



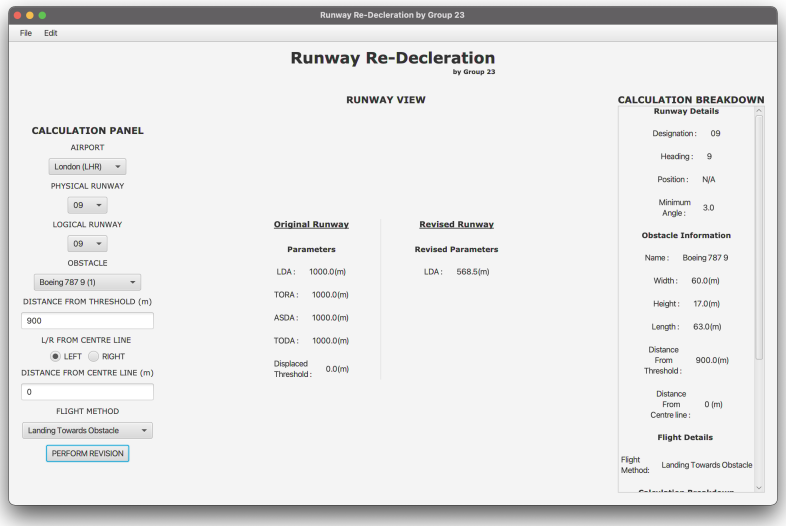
(a) Input error while performing a calculation



(b) Input error while adding a new runway

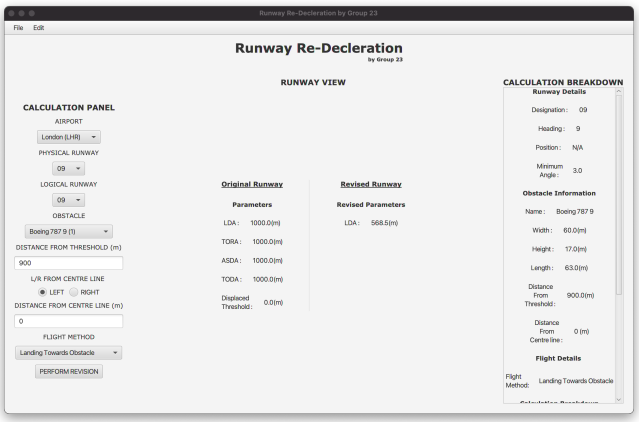
Runway Revision Calculations

Once the user has selected/provided the information for a runway revision within the 'Calculation Panel', the user can select the 'Perform Revision' button to be shown the updated runway parameters, as well as a side-by-side comparison of the updated parameters with the original. This information is displayed to the user in the 'Runway View', and the results of an example calculation are shown in the below screenshot.

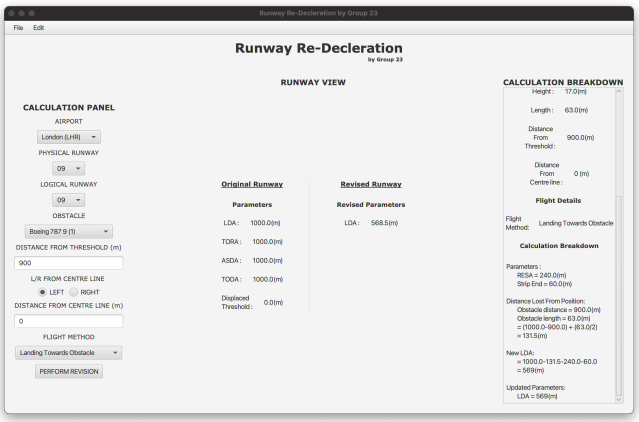


Runway Revision Calculation Breakdown

After a runway revision has been performed, the user is able to see a breakdown of the calculations that took place in the 'Calculation Breakdown' panel. The breakdown is contained within a scroll-able window, and provides the user with information on the runway being revised, the obstacle and its respective placement and a step-by-step description of the updated parameters were calculated. The below screenshots show the full contents of the calculation panel, following an example revision.

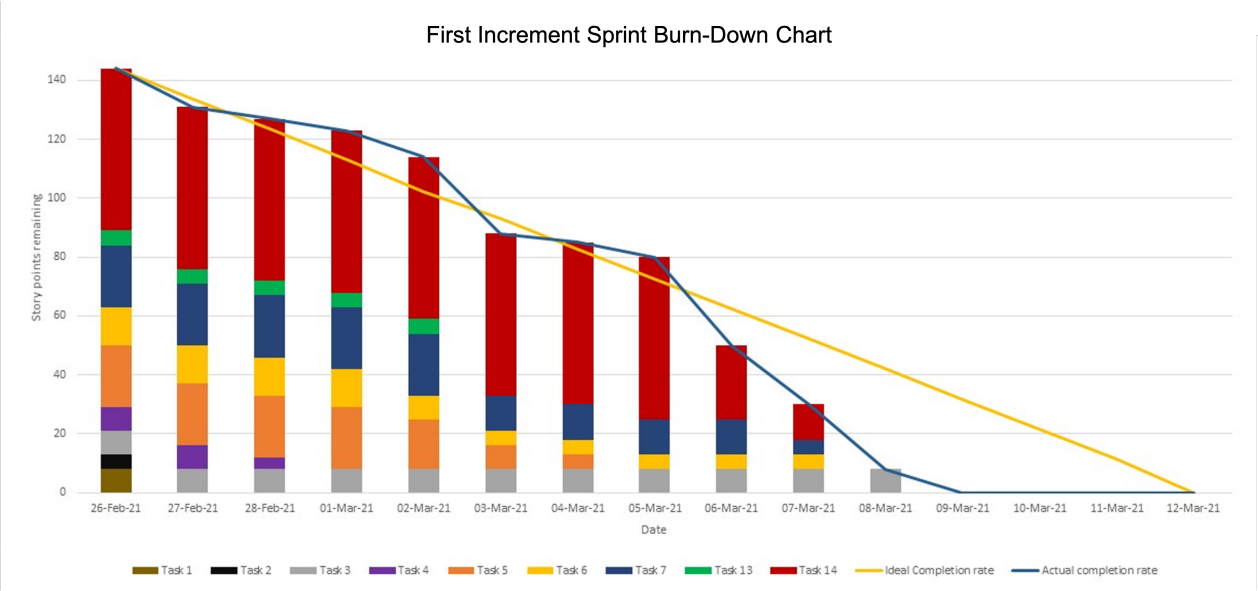


(a) First portion of the calculation breakdown



(b) Second portion of the calculation breakdown

B.3 Sprint Burn-Down Chart



C Supplementary Sprint 2 Information

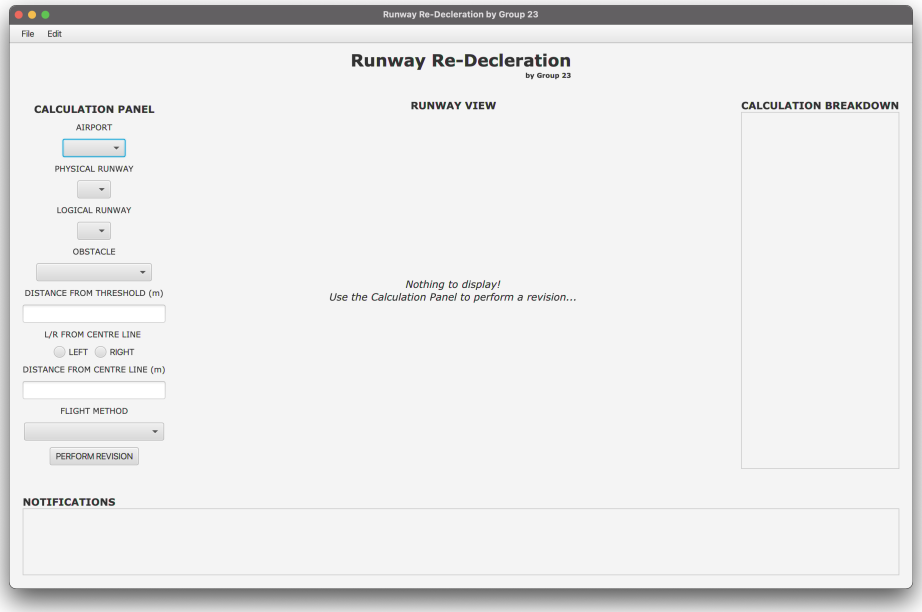
C.1 Sprint Report

`https://bit.ly/3eCW0TH`

C.2 Product Screenshots

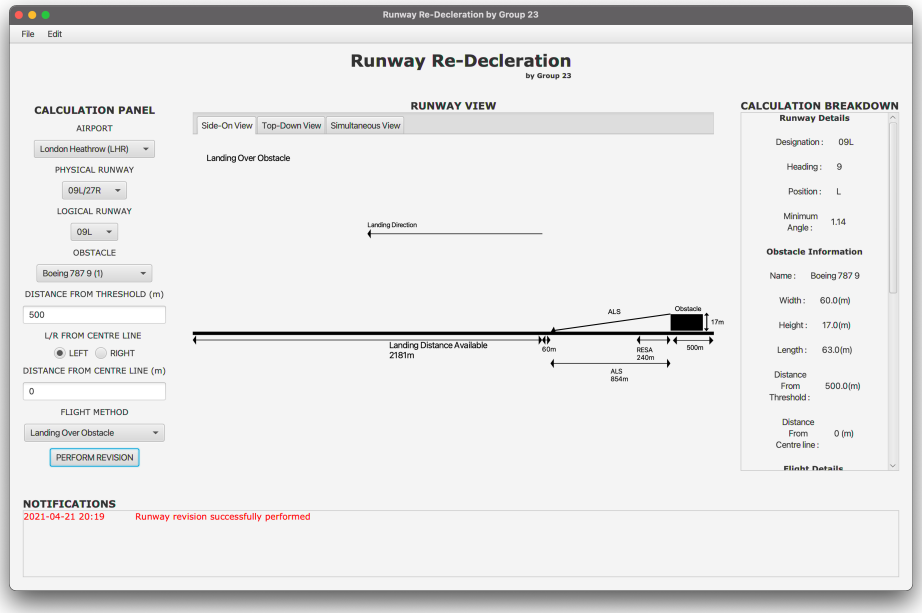
Dashboard

The 'Dashboard' functions as the home screen of the application, and has been updated to include a 'Notifications' panel (bottom) and support the graphical display of runway revisions. When a revision is performed, the 'Runway View' is converted to a tab pane that displays the results of the revision in a graphical form from both side-on and top-down perspectives. There is also a tab that allows the user to view both of these perspectives simultaneously.



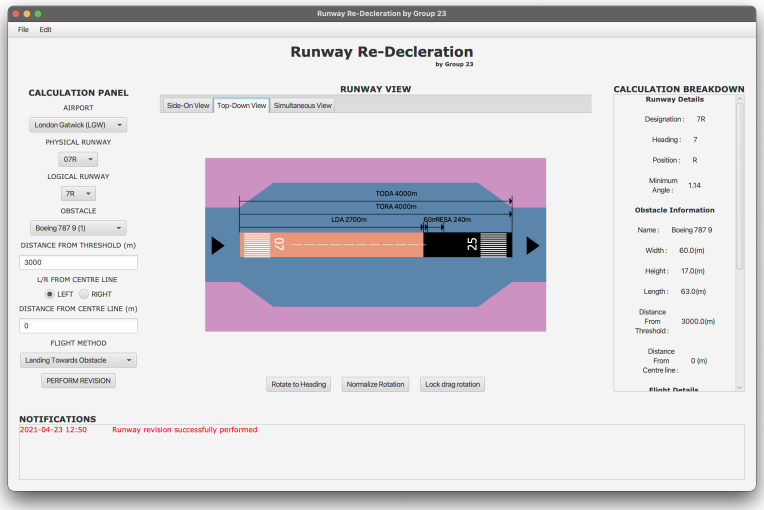
Viewing Revisions: Side-On Runway View

After performing a revision through the 'Calculation Panel', the user can view a side-on perspective of the revised runway within the 'Side-On' tab of the 'Runway View'. This perspective shows the updated runway parameters, the values that were involved in the calculation (including the ALS/TOCS) and the obstacle causing the re-declaration. The information is overlay-ed onto the image of the runway to provide the user with an appreciation of how the runway has been affected by the revision.



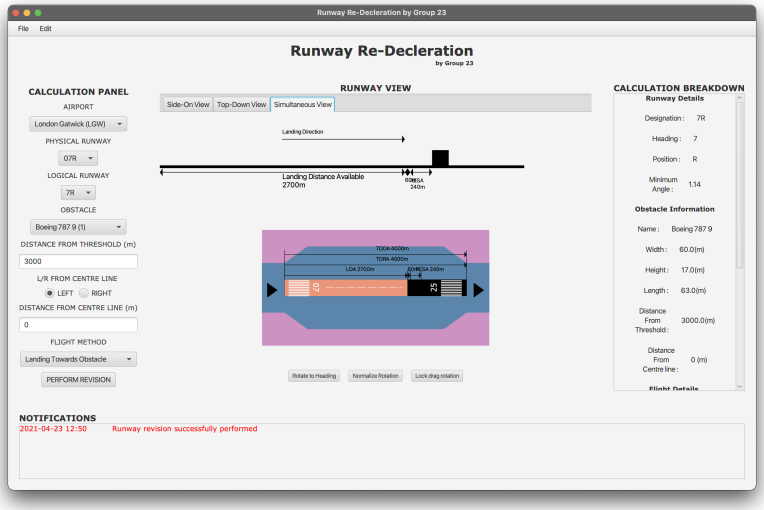
Viewing Revisions: Top-Down Runway View

After performing a revision through the 'Calculation Panel', the user can view a top-down perspective of the revised runway within the 'Top-Down' tab of the 'Runway View'. This perspective shows the updated runway parameters, the values that were involved in the calculation and the obstacle causing the re-declaration. The information is overlay-ed onto the image of the runway to provide the user with an appreciation of how the runway has been affected by the revision. The graphic also showcases the clear and graded area surrounding the runway.



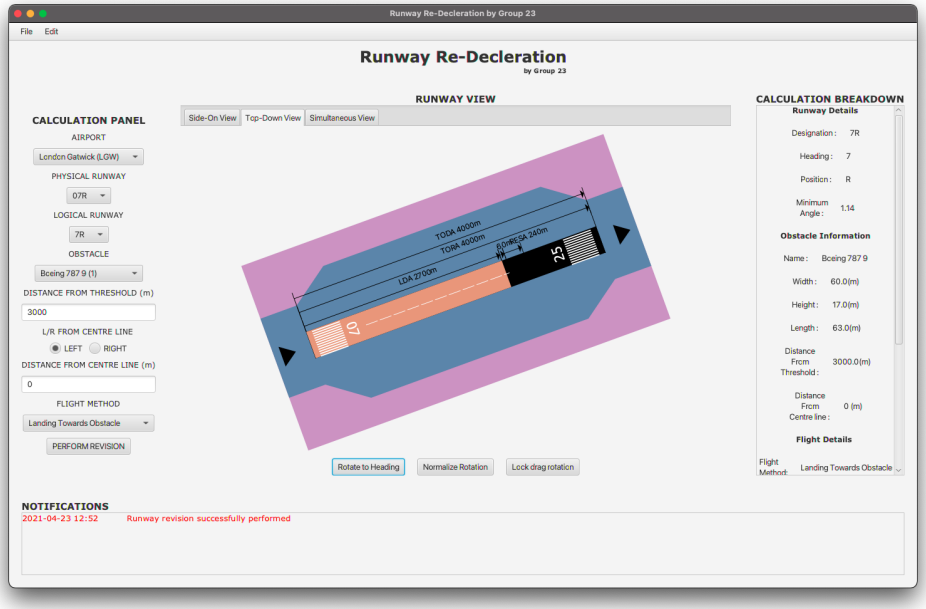
Viewing Revisions: Simultaneous Runway View

After performing a revision through the 'Calculation Panel', the user can navigate to the 'Simultaneous' tab within the 'Runway View' to be presented with a combination of both the top-down and side-on perspectives of the revision. The simultaneous view combines the displays from the 'Side-on' and 'Top-Down' tabs, presenting all of the information that is available in the individual sections.



Viewing Revisions: Rotating Top-Down Runway View

When viewing a revision inside the 'Top-Down' tab, the user is able to select the 'Rotate' button to rotate the graphic to match the heading of the runway currently being displayed. The system considers a heading of zero to be pointing North in the application (i.e., up) and all other headings can be derived from this basis. The 'Normalize' button can be used to 'un'-rotate the runway, back to its original position. The user is also able to drag the runway to rotate it manually.

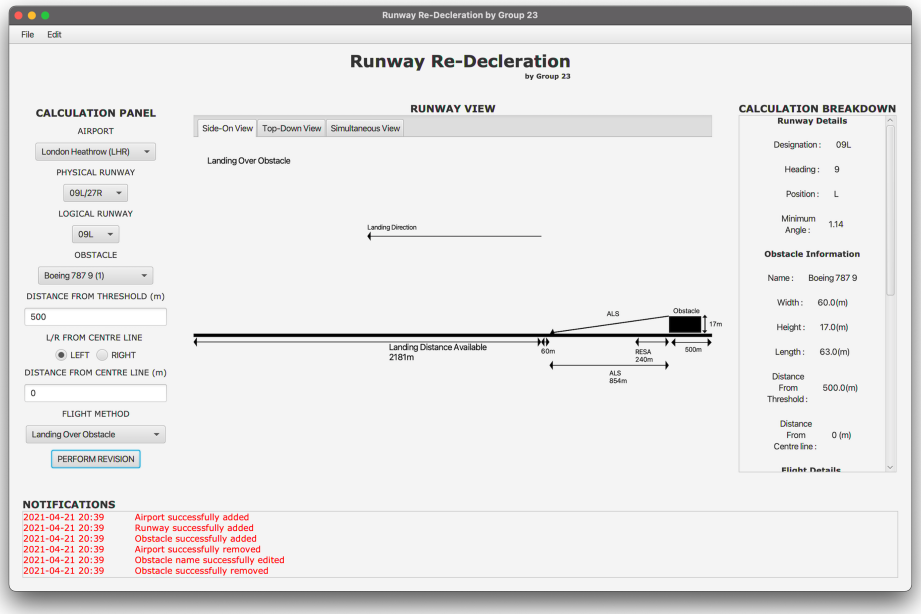


Notifications

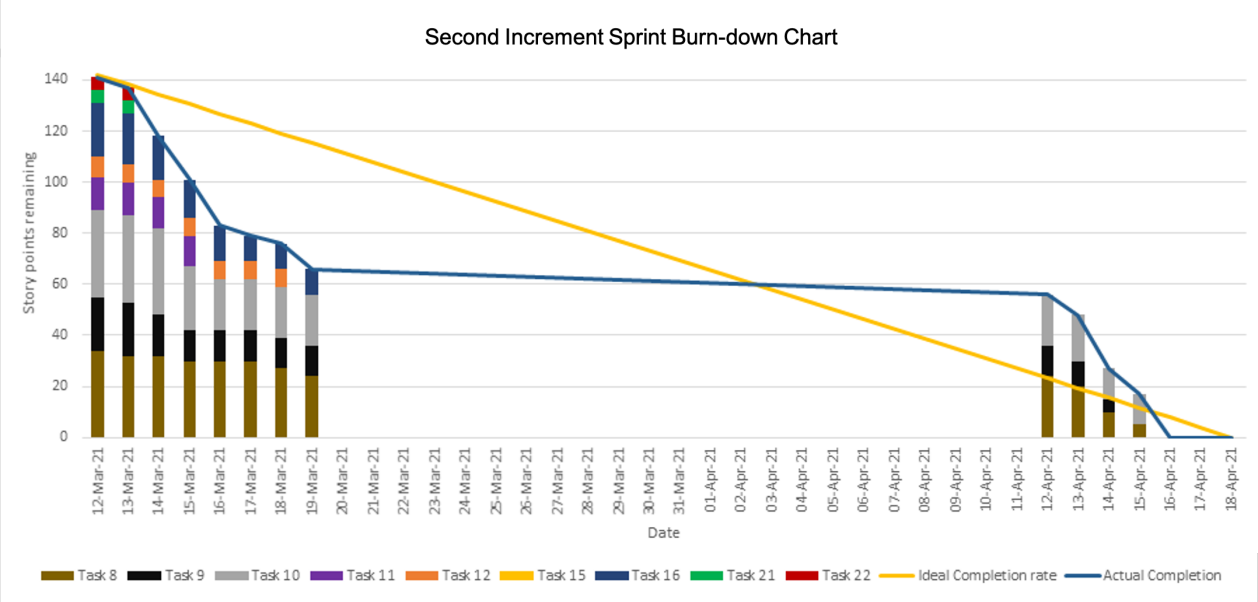
A notifications system has been implemented to provide the user with messages when interacting with the application. Each message is displayed to the user as text in the 'Notifications' panel of the dashboard, describing the action performed and providing a timestamp of that said action.

In addition to notifying the user when a revision has been performed (as per the product backlog items), several other actions will also trigger notifications. The full list of actions is described below.

- Placement of obstacle/Performing a revision
- Adding an airport to the system
- Adding an obstacle to the system
- Removing an airport from the system
- Removing an obstacle from the system
- Editing an airport in the system
- Editing an obstacle in the system



C.3 Sprint Burn-Down Chart



D Supplementary Sprint 3 Information

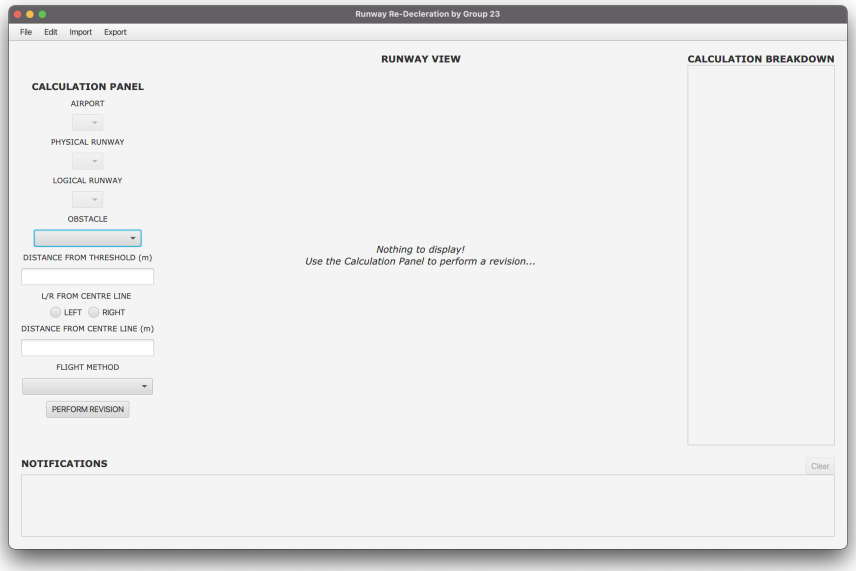
D.1 Sprint Report

`https://bit.ly/33zUeMI`

D.2 Product Screenshots

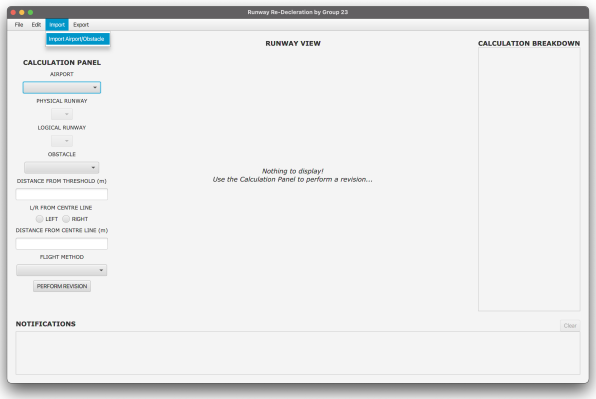
Dashboard

The 'Dashboard' functions as the home screen of the application, and has been updated to include 'Import' and 'Export' menus. These menu's can be found within the application toolbar. A button has also been added to the notification's panel that allows for the user to clear the currently stored notifications from the system (as the notifications now persist on application close).

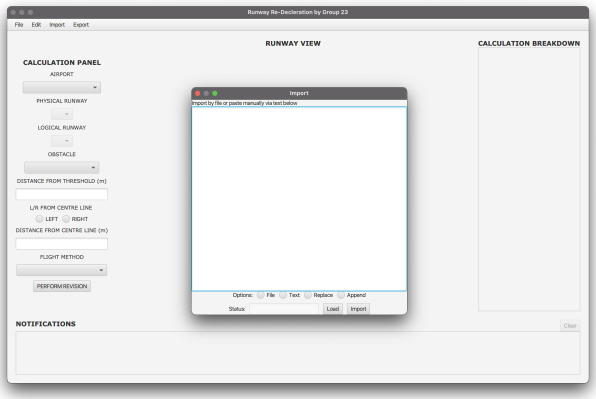


Importing Obstacles and Airports

By navigating to the 'Import' menu within the toolbar, the user is able to select to import an obstacle or airport. After selecting this menu item, a dialog is opened that allows for the user to either input the XML text for an airport/obstacle, or select an XML file to import into the system. Note that the user can import multiple obstacles/airports in one import operation. The dialog also provides the user with two loading options - "Replace" and "Append". "Replace" will replace the current system objects with the imported ones, whilst "Append" will append the imported objects to the existing ones.



(a) Import 'Airports/Obstacles' menu item within 'Import' menu

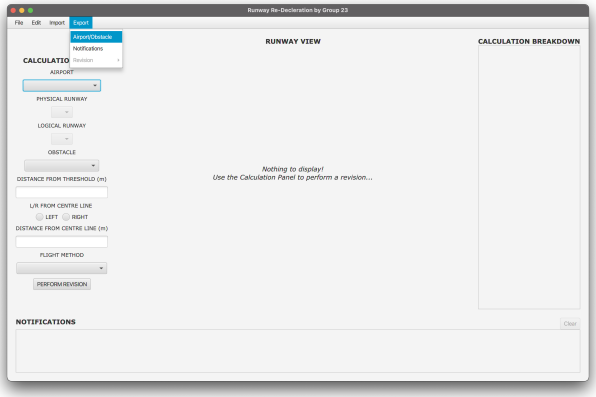


(b) The 'Import Airports/Obstacles' window

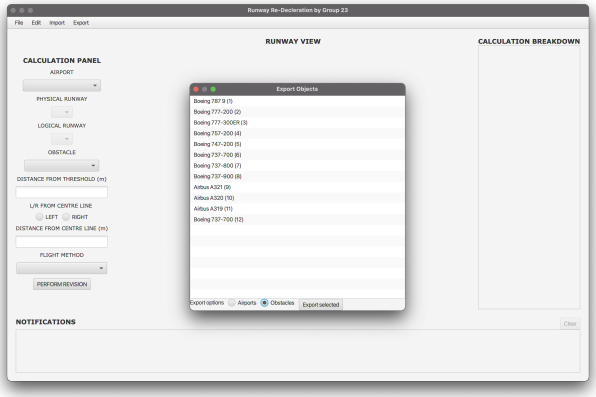
Exporting Obstacles and Airports

By navigating to the 'Export' menu within the toolbar, the user can select to export an obstacle or airport. After choosing this menu item, the user is presented with a dialog that allows for them to select the type of object to export (i.e., airport or obstacle), and which instances of this object type they would like to export. The user is able to select multiple objects to be exported at one time by holding the 'SHIFT' key whilst selecting.

Note that the format of the exported XML objects is the same format that is expected when importing objects from XML files, allowing for object instances to easily be transferred between different applications, or re-loaded in the case of a crash/from a backup.



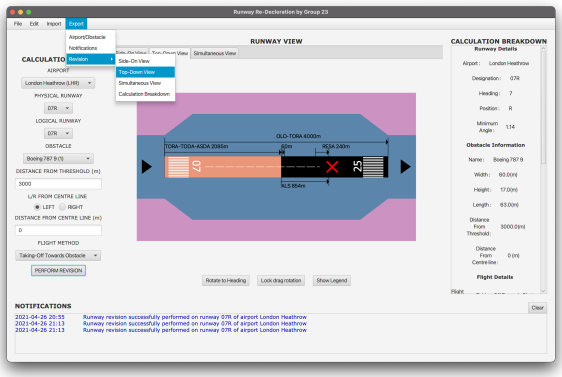
(a) Export 'Airports/Obstacles' menu item within 'Export' menu



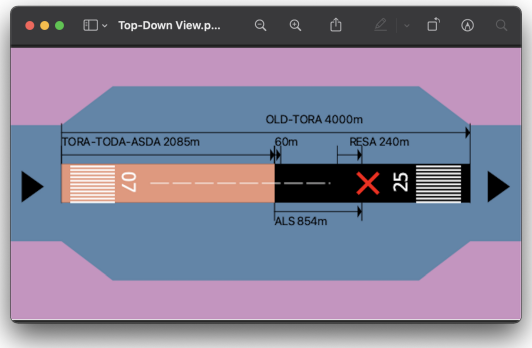
(b) The 'Export Airports/Obstacles' window

Exporting Runway Revision Graphics

After performing a revision, and after navigating to the 'Export' menu of the toolbar, the user is presented with an 'Export Revision' sub-menu that allows for them to export the results of the revision outside of the application. The user can chose to export the side-on view, top-down view, or simultaneous view of the revision, and can export the chosen view as a JPEG, PNG or GIF file. A file-chooser dialog is displayed to allow the user to select the location of the exported file.



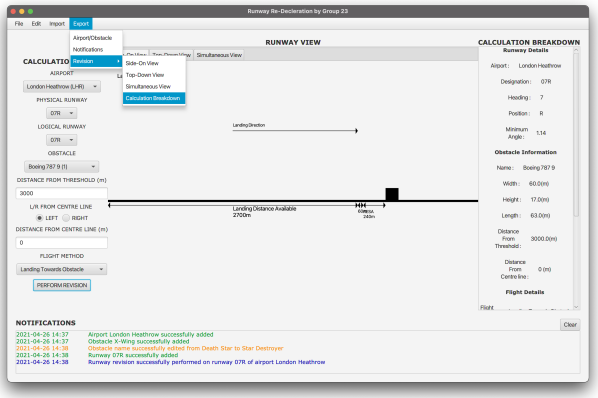
(a) Export 'Revisions' menu item within 'Export' menu



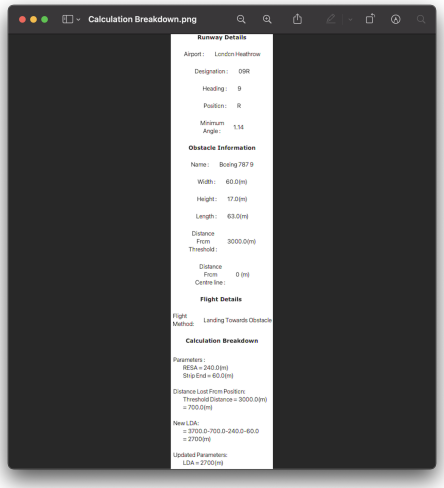
(b) The top-down view exported from the application

Exporting Runway Revision Calculation Breakdown

After performing a revision, and after navigating to the 'Export' menu of the toolbar, the user is presented with an 'Export Revision' sub-menu that allows for them to export the calculation breakdown of the revision outside of the application. The user can chose to export the calculation breakdown as a PNG, JPEG or GIF file. A file-chooser dialog is displayed to allow the user to select the location of the exported file. This exported calculation breakdown image can then be printed by the user.



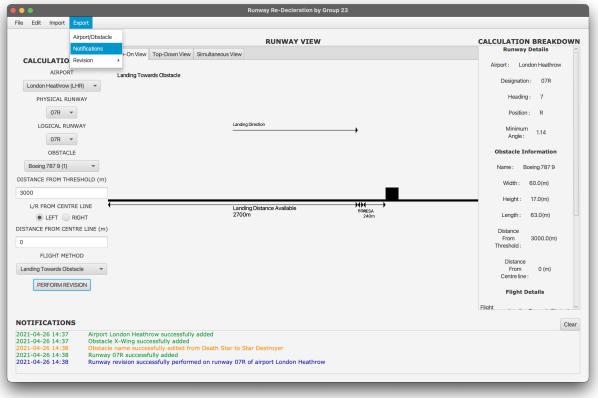
(a) Export 'Revisions' menu item within 'Export' menu



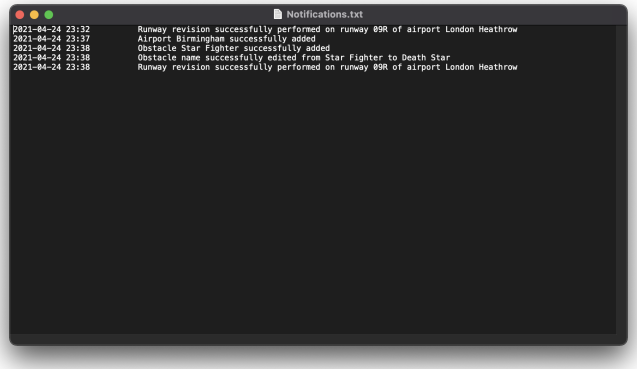
(b) The calculation breakdown exported from the application

Exporting System Notifications

When notifications are present in the system, and after navigating to the 'Export' menu within the toolbar, the user can select to export the system notifications. The notifications are then exported as a text file into the chosen location.



(a) Export 'Notifications' menu item within 'Export' menu



(b) The notifications exported from the application

D.3 Sprint Burn-Down Chart