FIT3162
User Guide

Designing software effort estimation model using machine learning techniques

Jovan Ong Shung Jiet
29274311
jong0016@student.monash.edu

Benjamin Marc Wijayaratne
30955246
bwij0006@student.monash.edu

Wong Zi Qi
30897548
zwon0015@student.monash.edu

Team: MCS17
Supervisor: Dr Golnoush Abaei
Word count: 1913

Table of contents

End User Guide

This user manual will demonstrate how to estimate the effort required to develop a new software project using our web application. Users are expected to have a decent knowledge of professional software terminology.

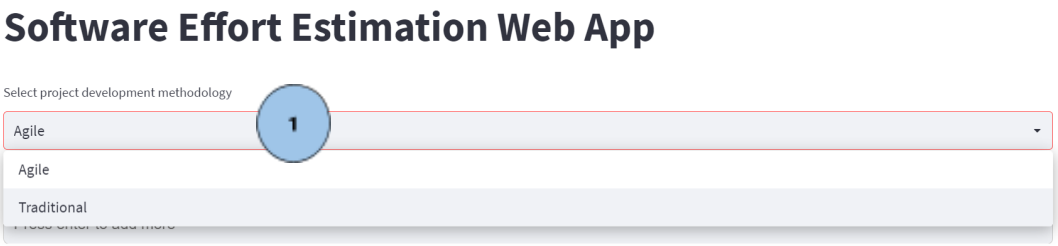
1. Getting started

1.1 Getting access to our Web Application

Navigate to here using web browser.

1.2 Selecting project development methodology

Users must select the software development methodology that will work best for the project. Teams using the Agile approach develop software in iterations that introduce new functionality in small increments. Teams employing the traditional (waterfall) method develop projects in sequential phases, often with clear objectives and stable requirements.



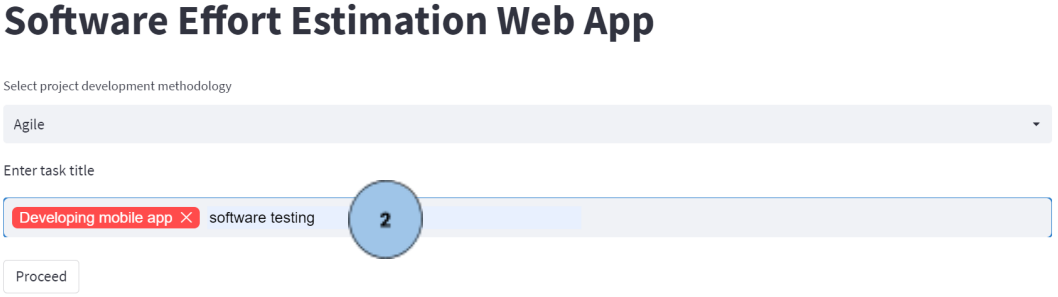
2. Filling in form

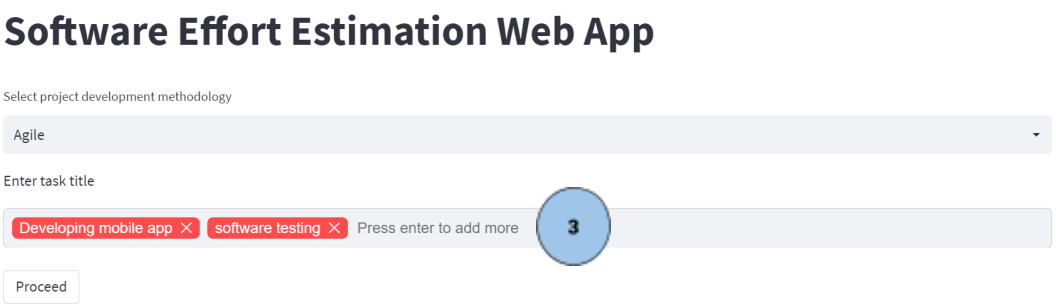
2.1 Agile approach

The following will demonstrate the input instructions for Agile selection.

2.1.1 Task titles

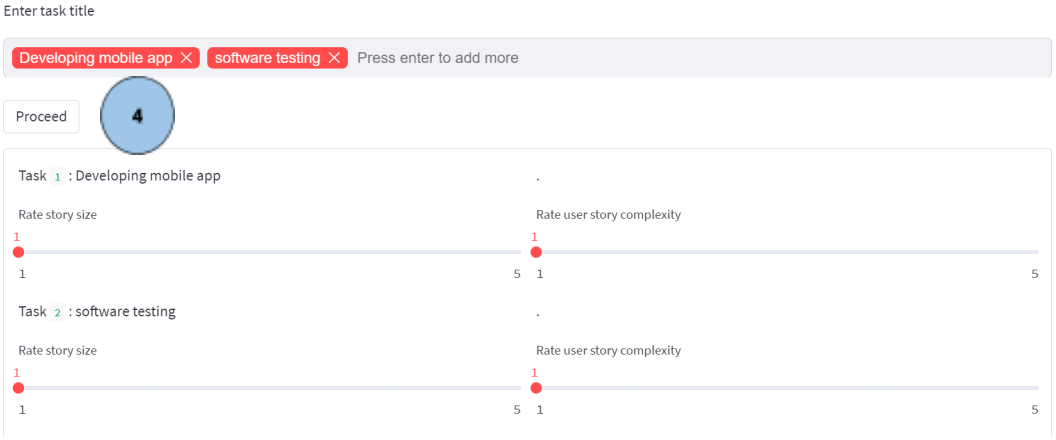
Users must key in one or many task titles. Press enter to add more titles.





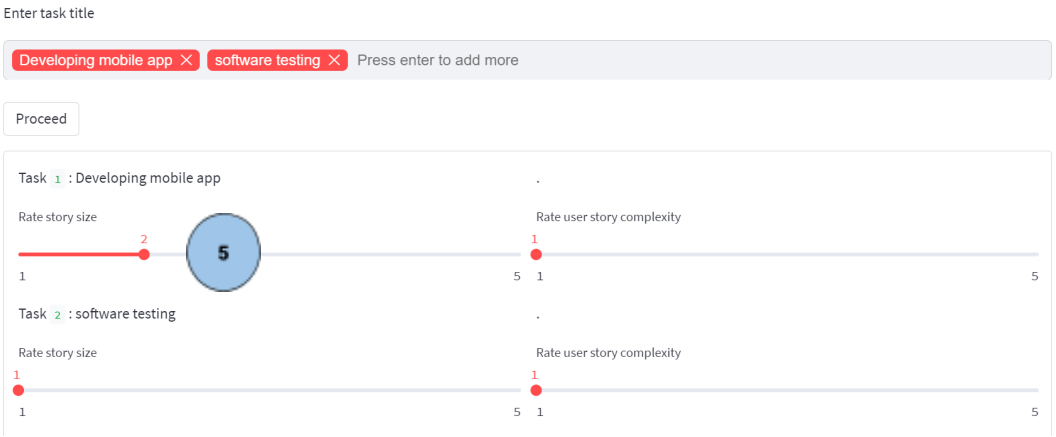
2.1.2 User Story

Users are required to click the Proceed button to generate a number of two different scales that helps to determine the effort of a particular user story based on the number of tasks given. A user story refers to the smallest unit of work in an agile framework.



2.1.3 Story size scale

Users are required to rate the estimated size of the relative scale of the work in terms of actual development effort for each task, following the provided scale. *Table 1* shows the scale assigns five values to different types of user stories according to their size.

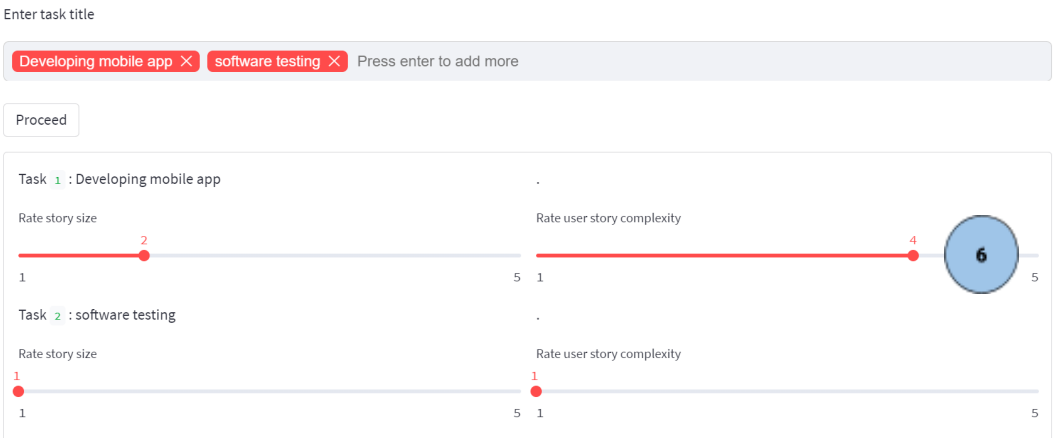


Values	Guidelines
5	• An extremely large story • Too large to accurately estimate • Should almost certainly be broken down into a set of smaller Stories • May be a candidate for separation into a new project
4	• A very large Story • Requires the focused effort of a developer for a long period of time – Think in terms of more than a week of work • Should consider breaking it down into a set of smaller stories
3	• A moderately large story • Think in terms of two to five days of work
2	• Think in terms of a roughly a day or two of work
1	• A very small story representing tiny effort level • Think in terms of only a few hours of work.

Table 1: Story Size Scale (Zia et al., 2012)

2.1.4 User story complexity scale

Users are required to rate the complexity in terms of the requirements of the story and technical complexity, following the provided scale. *Table 2* shows the scale that assigns five values to user stories according to their nature.



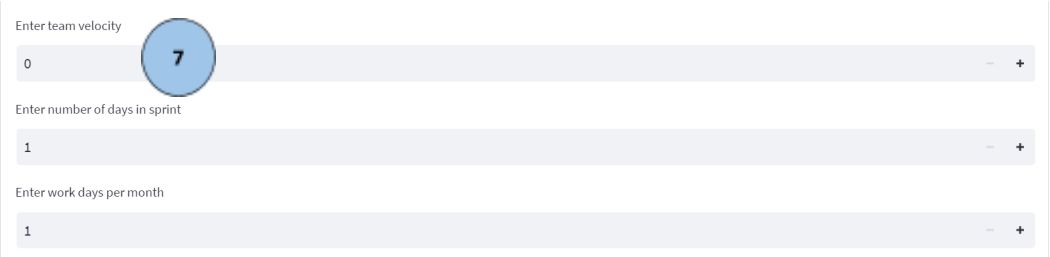
Value	Guidelines
5	● Extremely complex ● Many dependencies on other stories, other systems or subsystems ● Represents a skill set or experience that is important, but absent in the team ● Story is difficult to accurately describe ● Many unknowns ● Requires significant refactoring ● Requires extensive research ● Requires difficult judgment calls ● Effects of the Story have significant impact external to the story itself
4	● Very complex ● Multiple dependencies on other stories, other systems or subsystems ● Represents a skill set or experience that is important, but not strong in the team ● Story is somewhat difficult for product owner to accurately describe ● Multiple unknowns ● Comparatively large amount of refactoring required ● Requires research ● Requires senior level programming skills to complete ● Requires somewhat difficult judgment calls ● Effects of the Story have moderate impact external to the story itself
3	● Moderately complex ● Moderate number of dependencies on other stories, other systems or subsystems ● Represents a skill set or experience that is reasonably strong in the team ● Story is somewhat difficult for owner to accurately describe ● Moderate level of unknowns ● Some refactoring may be required ● Requires intermediate programming skills to complete ● Requires little research ● Requires few important judgment calls ● Effects of the Story have minimal impact external to the story itself
2	● Easily understood technical and business requirements ● Little or no research required ● Few unknowns ● Little if any research required ● Requires basic to intermediate programming skills to complete ● Effects of the Story are almost completely localized to the Story

	itself
1	• Very straightforward with few if any unknowns • Technical and business requirements very clear with no ambiguity • No unknowns • No research required • Requires basic programming skills to complete • Effects of Story are completely localized to the Story itself

Table 2: Complexity Scale (Zia et al., 2012)

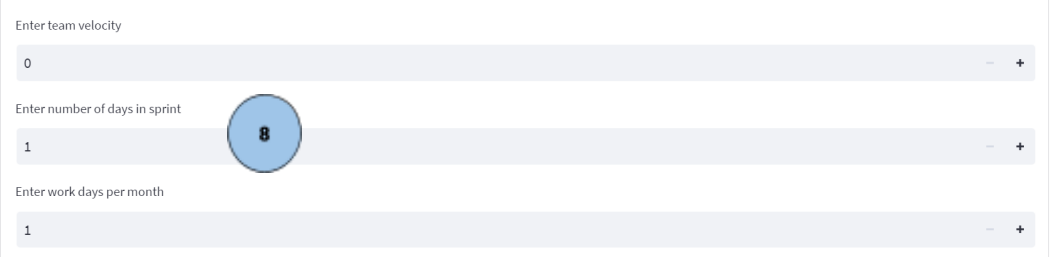
2.1.5 Team Velocity

Users are required to enter how many units of effort the team can complete in a typical sprint, or in other words, the total story points the team can complete per sprint. The minimum value of team velocity is set to zero.



2.1.6 Sprint Size

Users are required to enter the sprint duration that the team needs to work on and complete user stories. The sprint length should be between 1 and 4 weeks, which is 1 and 28 days. Agile scrum project management is never used for timelines longer than 4 weeks per sprint.

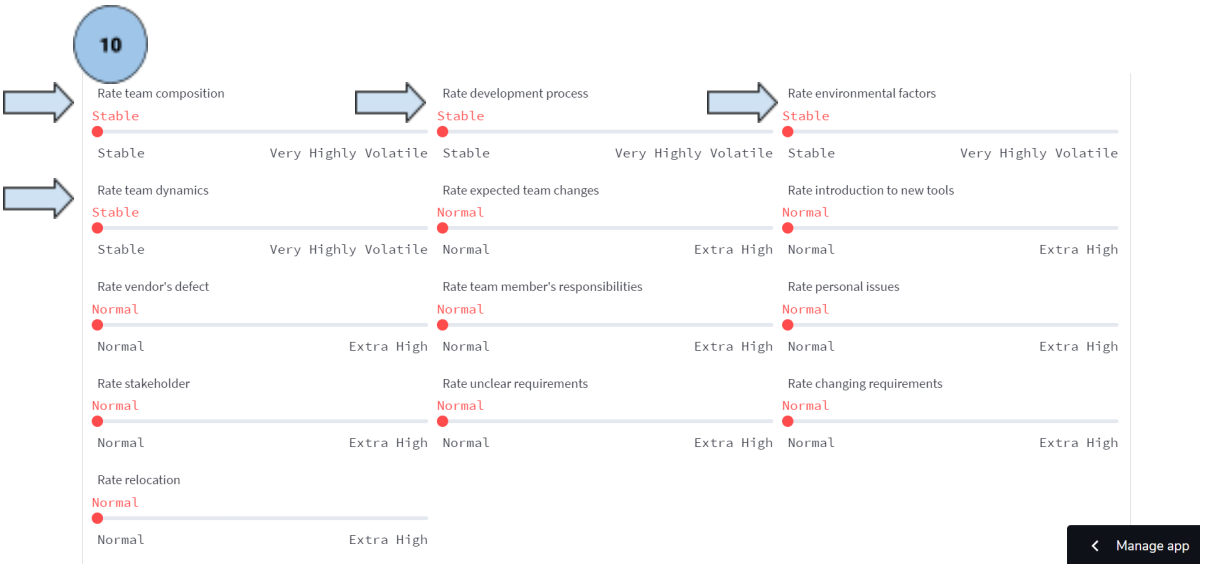


2.1.7 Work days

Users are required to enter the number of working days per month. The work days should be between 1 and 31 days.

2.1.8 Friction Factors

Users are required to rate friction factors that lead to a constant drag on productivity and thus, a reduction in project velocity. These factors include Team Composition – the extent of having the right people with the right skills on the team, Process – the modifications to project development processes, i.e. Agile methods, building, releasing, testing, etc…, Environmental Factors – interruptions, unpleasant seating and desks, insufficient hardware or software, noise disturbance, poor ventilation, poor lighting, etc…, and Team Dynamics – the behavioral relationships between team members. *Table 3* displays four friction factors with a range of values. These values have been modified in accordance with the degree of risk involved. The overall friction will be computed as the product of all four friction factors.

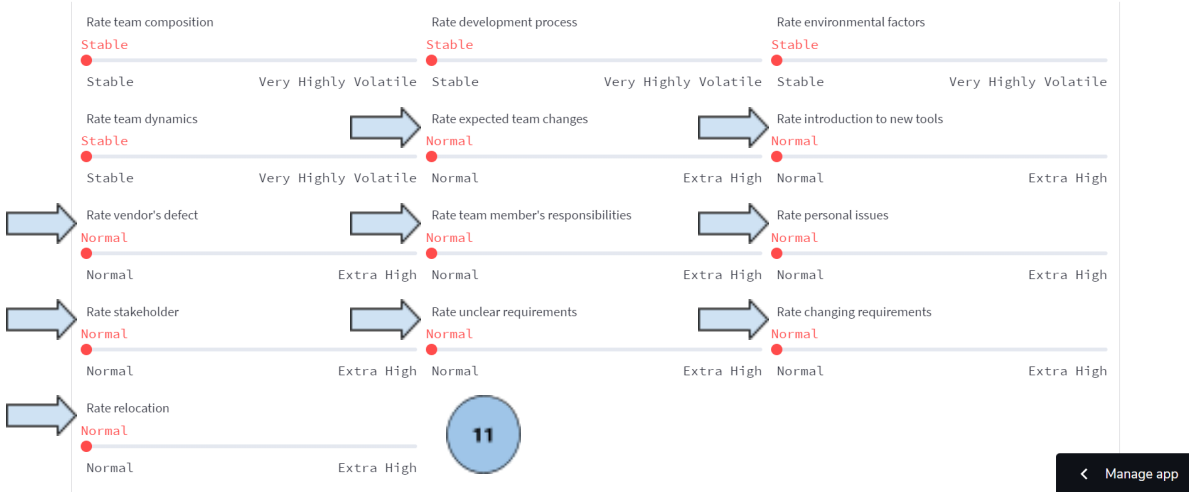


Friction factor	Stable	Volatile	Highly Volatile	Very Highly Volatile
Team Composition	1	0.98	0.95	0.91
Process	1	0.98	0.94	0.89
Environmental Factors	1	0.99	0.98	0.96
Team Dynamics	1	0.98	0.91	0.85

Table 3: Friction Factors (Zia et al., 2012)

2.1.9 Dynamic Force Factors

Users are required to rate dynamic forces factors that decelerate the project development and lead to a loss in velocity due to its unforeseeable and unexpected nature. These factors include Team Changes – the extent of changes happening on the team, i.e. adding or removing members, adjusting roles and responsibilities, New Tools – introducing new programming language, development tools, database systems, etc… which require learning time until mastered, Vendor Defects – third-party tool and software flaws that necessitate developer workarounds, Responsibilities outside of the project – team members taking on additional duties that aren't related to the project, Personal Issues – personal health, family dynamics, etc…, Stakeholders – stakeholders being unresponsive when the developers or tester asks them for information, causing delays, or they could have unrealistic expectations for the group, Unclear Requirements – insufficient clarity or details in requirements that might cause unnecessary rework, Changing Requirements – new project specifications demand abilities that the team lacks or isn't strong at, and Relocation – moving the team to a new location which might disrupt the work rhythm. *Table 4* displays nine dynamic forces factors with a range of values. These values have been modified in accordance with the degree of risk involved. The overall dynamic forces will be computed as the product of all nine dynamic forces factors.



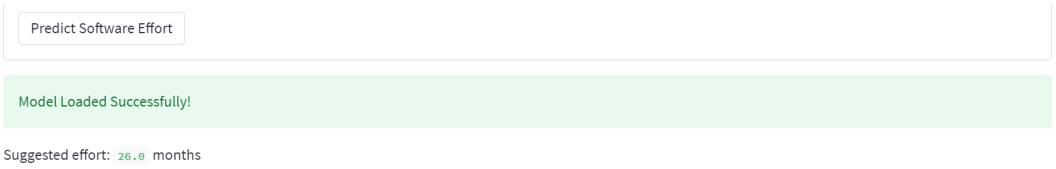
Variable Factor	Normal	High	Very High	Extra High
Expected Team Changes	1	0.98	0.95	0.91
Introduction of New Tools	1	0.99	0.97	0.96
Vendor's Defect	1	0.98	0.94	0.90
Team member's responsibilities outside the project	1	0.99	0.98	0.98
Personal Issues	1	0.99	0.99	0.98
Expected Delay in Stakeholder response	1	0.99	0.98	0.96
Expected Ambiguity in Details	1	0.98	0.97	0.95
Expected Changes in environment	1	0.99	0.98	0.97
Expected Relocation	1	0.99	0.99	0.98

Table 4: Dynamic Forces Factors (Zia et al., 2012)

2.1.10 Software Effort Prediction

Users are required to click the Predict Software Effort button to compute the duration needed to complete the project. The inputs to our model consist of the Sum of efforts of all individual user stories in story points, Initial velocity, Deceleration, Final velocity, Sprint size, and Work days. The Sum of efforts is computed as the sum of the product of the story size and complexity. Dividing team velocity with sprint size gives Initial velocity. Declaration is calculated by multiplying friction with dynamic forces. Initial velocity is optimized to Final velocity by considering the deceleration value using the formula $V = (V_i)^D$. The estimated effort in days will then be converted to months by dividing Work days.

Output Example:

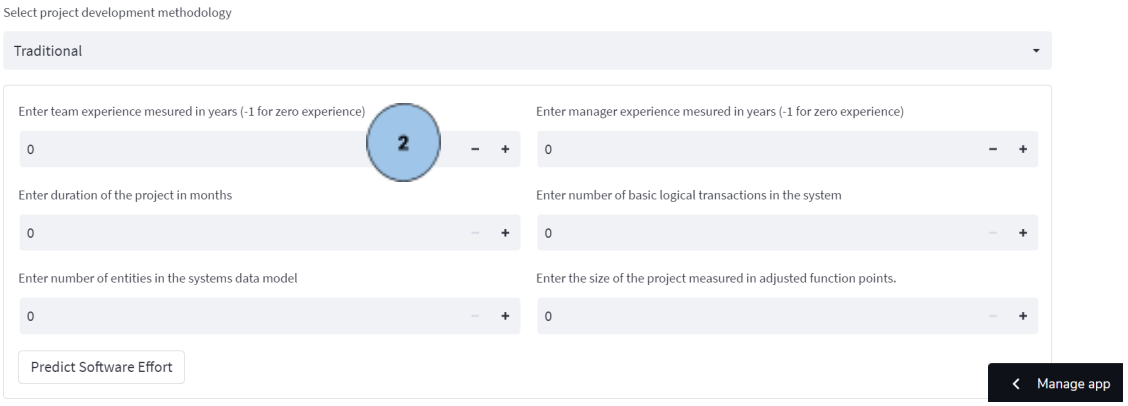


2.2 Traditional Approach

The following will demonstrate the input instructions for traditional selection.

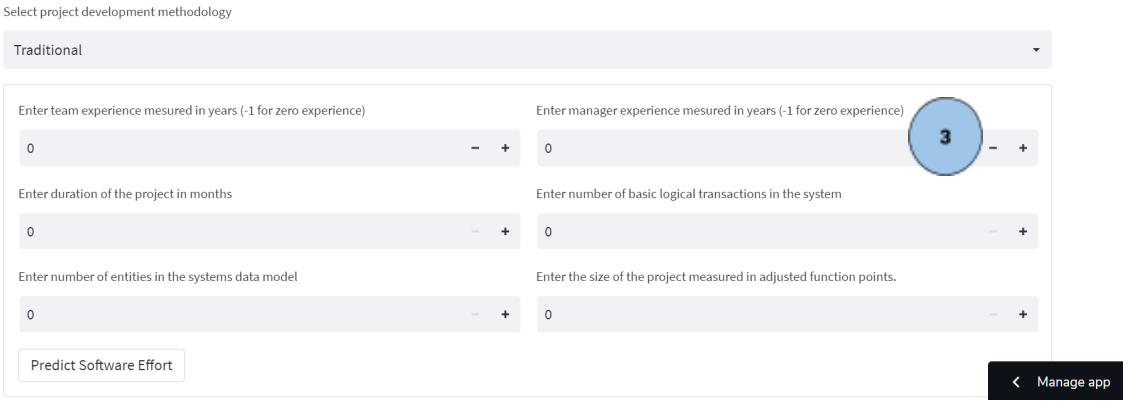
2.2.1 Team Experience

Users are required to enter the team experience measured in years. The minimum value is set to -1 which indicates no experience, while 0 indicates that the team has months of experience but has yet to reach a year.



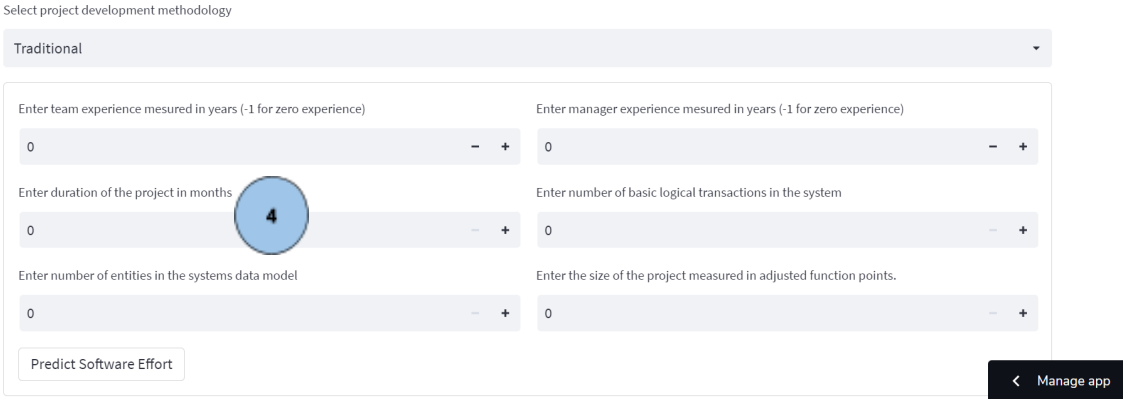
2.2.2 Manager Experience

Users are required to enter the experience of the project manager measured in years. The minimum value is set to -1 which indicates no experience, while 0 indicates that the manager has months of experience but has yet to reach a year.



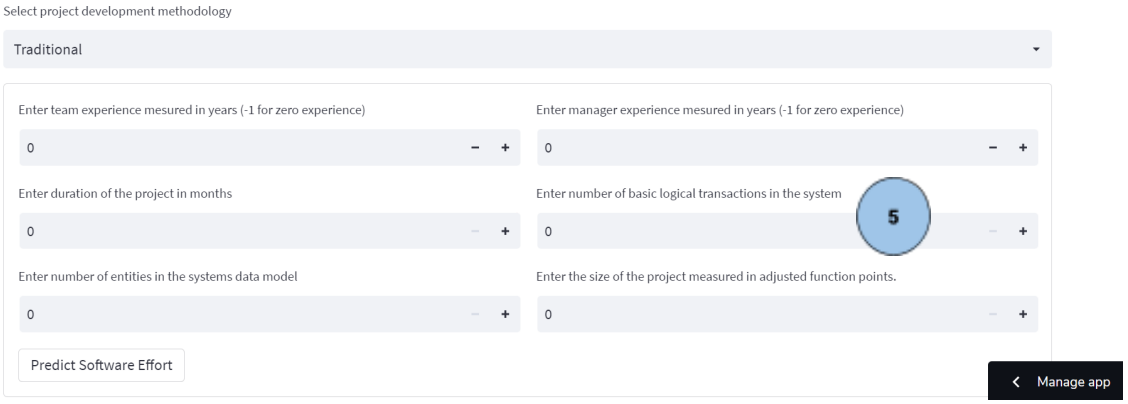
2.2.3 Project Duration

Users are required to enter the expected duration of the project in months.



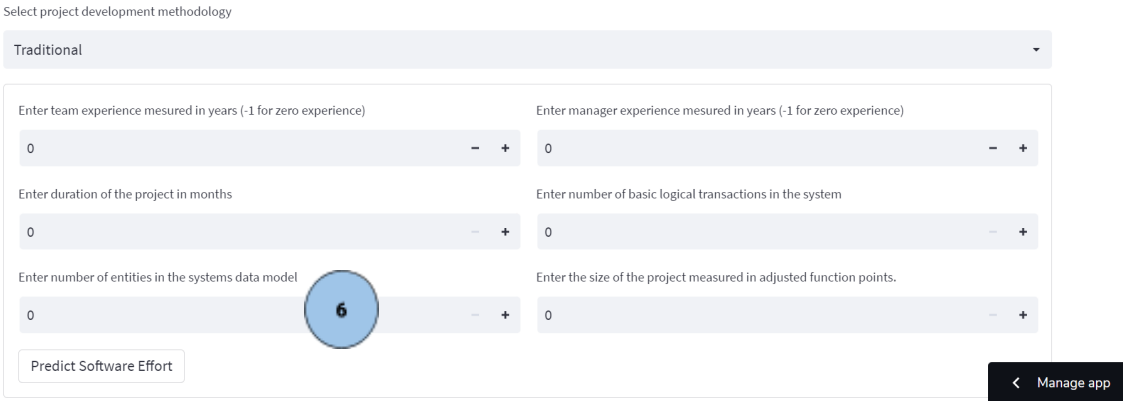
2.2.4 Logical Transactions

Users are required to enter the expected count of basic logical transactions in the database system. The planned project objectives will provide an indication of the expected logical transaction required to develop the project in the database.



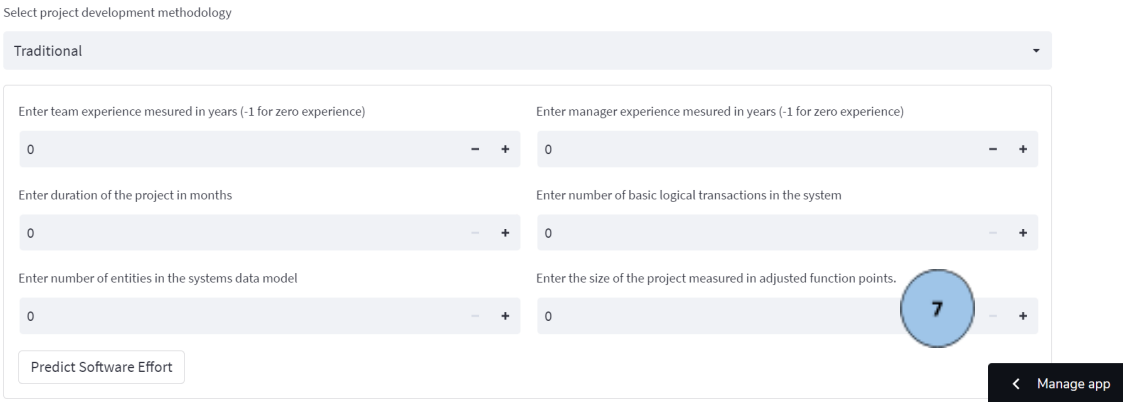
2.2.5 Entities

Users are required to enter the expected number of entities in the systems data model. The planned project objectives will provide an indication of the expected structure of data involved in the project.



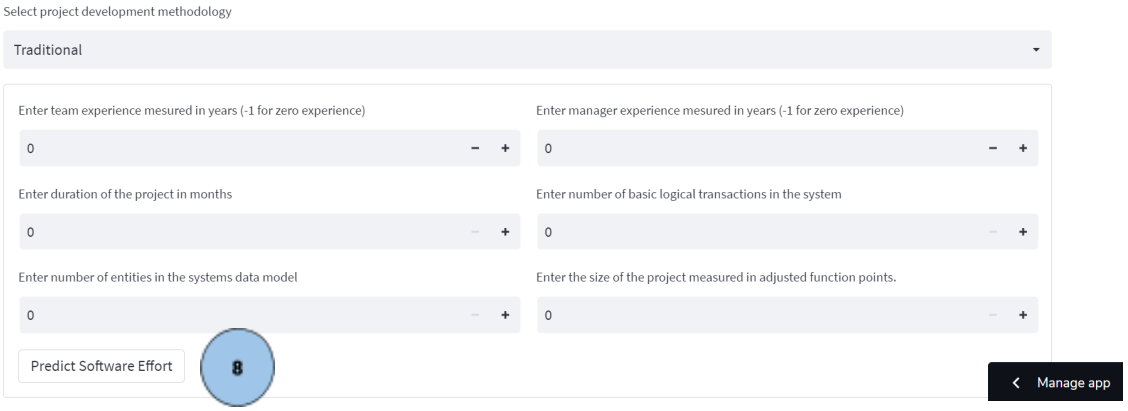
2.2.6 Project Size

Users are required to enter the size of the project measured in adjusted function points. This measurement is used for solving software requirements based on the different functions that the requirement can be divided into.

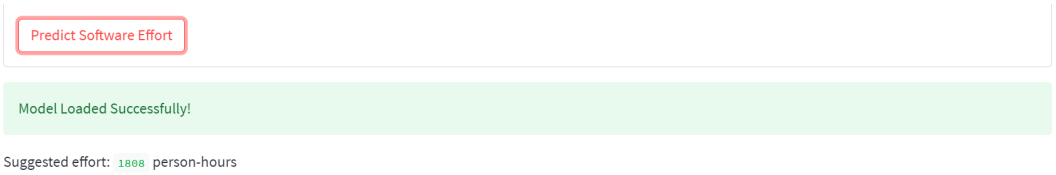


2.2.7 Software Effort Prediction

Users are required to click the Predict Software Effort button to compute the actual effort to complete the project measured in person-hours. The inputs to our model consist of all the input fields described above.



Output Example:



Technical Guide

This technical guide will demonstrate how to replicate and run our web application locally. The source code can be found here.

1. Prerequisites

1.1 Install Anaconda

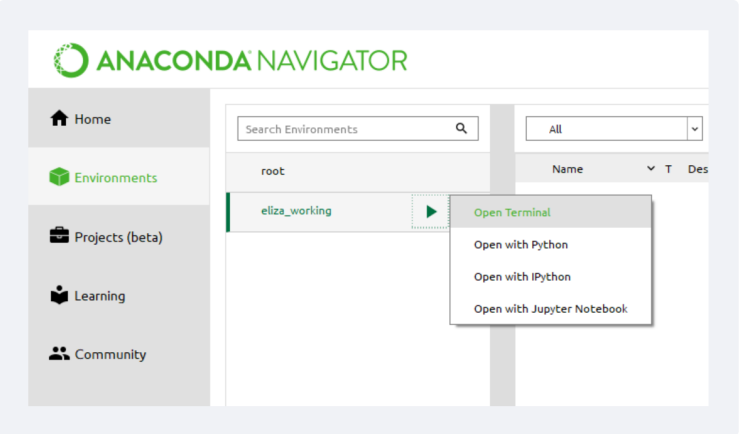
If you don't have Anaconda Navigator installed yet, follow the steps provided here.

1.2 Setting up a new environment

Create and manage a new environment using Anaconda Navigator.

1.3 Install Dependencies

In Anaconda Navigator, open the terminal in your working environment:



In the terminal that appears, type:

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

2. About the Models

The trained models are saved as .joblib file. Use joblib.load to access the models.

3. About the Training Process

The model training scripts can be found in .ipynb file.

4. About the Web Application

To access the web app locally, open the terminal in your working environment in Anaconda Navigator and type:

```
streamlit run app.py
```

Or access the web app here to interact with our trained models using web browser.