Software Requirements Specification for 4TB6 - Mechatronics Capstone: subtitle describing software

Author Name(s)

October 4, 2022

Contents

Revision History

Date	Version	Notes
Date 1	1.0	Notes
Date 2	1.1	Notes

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

Throughout this document SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	**unit**	**SI**
m	length	metre
kg	mass	kilogram
s	time	second
°C	temperature	centigrade
J	energy	joule
W	power	watt ($\mathrm{W} = \mathrm{J\,s^{-1}}$)

[Only include the units that your SRS actually uses. —TPLT]

[Derived units, like newtons, pascal, etc, should show their derivation (the units they are derived from) if their constituent units are in the table of units (that is, if the units they are derived from are used in the document). For instance, the derivation of pascals as $\mathrm{Pa} = \mathrm{N\,m^{-2}}$ is shown if newtons and m are both in the table. The derivations of newtons would not be shown if kg and s are not both in the table. —TPLT]

[The symbol for units named after people use capital letters, but the name of the unit itself uses lower case. For instance, pascals use the symbol Pa, watts use the symbol W, teslas use the symbol T, newtons use the symbol N, etc. The one exception to this is degree Celsius. Details on writing metric units can be found on the NIST web-page. —TPLT]

1.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with the heat transfer literature and with existing documentation for solar water heating systems. The symbols are listed in alphabetical order.

symbol	**unit**	**description**
A_C	$\mathrm{m^2}$	coil surface area
A_{in}	$\mathrm{m^2}$	surface area over which heat is transferred in

[Use your problems actual symbols. The si package is a good idea to use for units. —TPLT]

1.3 Abbreviations and Acronyms

symbol	description
A	Assumption
DD	Data Definition
GD	General Definition
GS	Goal Statement
IM	Instance Model
LC	Likely Change
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
4TB6 - Mechatronics Capstone	[put an expanded version of your program name here (as appropriate)
T	Theoretical Model

[Add any other abbreviations or acronyms that you add —TPLT]

1.4 Mathematical Notation

[This section is optional, but should be included for projects that make use of notation to convey mathematical information. For instance, if typographic conventions (like bold face font) are used to distinguish matrices, this should be stated here. If symbols are used to show mathematical operations, these should be summarized here. In some cases the easiest way to summarize the notation is to point to a text or other source that explains the notation. —TPLT]

[This section was added to the template because some students use very domain specific notation. This notation will not be readily understandable to people outside of your domain. It should be explained. —TPLT]

[This SRS template is based on **??**. It will get you started. You should not modify the section headings, without first discussing the change with the course instructor. Modification means you are not following the template, which loses some of the advantage of a template, especially standardization. Although the bits shown below do not include type information, you may need to add this information for your problem. If you are unsure, please can ask the instructor. —TPLT]

[Feel free to change the appearance of the report by modifying the LaTeX commands. —TPLT]

[This template document assumes that a single program is being documented. If you are documenting a family of models, you should start with a commonality analysis. A separate template is provided for this. For program families you should look at **??**. Single family member programs are often programs based on a single physical model. General purpose tools are usually documented as a family. Families of physical models also come up. —TPLT]

[The SRS is not generally written, or read, sequentially. The SRS is a reference document. It is generally read in an ad hoc order, as the need arises. For writing an SRS, and for reading one for the first time, the suggested order of sections is:

- Goal Statement
- Instance Models
- Requirements
- Introduction
- Specific System Description

—TPLT]

[Guiding principles for the SRS document:

- Do not repeat the same information at the same abstraction level. If information is repeated, the repetition should be at a different abstraction level. For instance, there will be overlap between the scope section and the assumptions, but the scope section will not go into as much detail as the assumptions section.

—TPLT]

[The template description comments should be disabled before submitting this document for grading. —TPLT]

[You can borrow any wording from the text given in the template. It is part of the template, and not considered an instance of academic integrity. Of course, you need to cite the source of the template. —TPLT]

[When the documentation is done, it should be possible to trace back to the source of every piece of information. Some information will come from external sources, like terminology. Other information will be derived, like General Definitions. —TPLT]

[An SRS document should have the following qualities: unambiguous, consistent, complete, validatable, abstract and traceable. —TPLT]

[The overall goal of the SRS is that someone that meets the Characteristics of the Intended Reader (Section 2.3) can learn, understand and verify the captured domain knowledge. They should not have to trust the authors of the SRS on any statements. They should be able to independently verify/derive every statement made. —TPLT]

2 Introduction

The purpose of the Smart Lock project is to design and build a product that will provide bicycle users with a safer, easier, and more accessible way to secure their bike through their smartphone. Additionally, it will provide users with a GPS feature to locate the lock in case of bike theft or misplacement. It will consist of a physical lock that mounts to a bike and a smartphone application that will function as the user interface through which the lock can be engaged and disengaged wirelessly, as well as located. The project will provide an engineering solution using wireless communication, mechanical design, and smartphone application development. More broadly, it seeks to encourage members of society to pursue biking, in both a transportation and recreational capacity, improving the health of society's citizens and its environment.

2.1 Purpose of Document

The Requirements Documentation seeks to provide a complete overview of the requirements of the Smart Lock system so as to define the project. The assumptions, inputs and outputs will also be defined in order to outline the problem. Several models will be developed and lastly, likely and unlikely changes will be described. This will communicate a unified and documented plan for the project that can be used after the design stage to verify its functionality and provide an important benchmark.

2.2 Scope of Requirements

The Smart Lock project will build off existing designs to tie together principles of wireless communication, GPS, a mechanical locking mechanism, electrical actuation, and smartphone application development. The system will provide all functions necessary for effective and secure locking and location.

The Smart Lock project will be designed to perform the following functions for the user.

1. Wireless communication from the smartphone to the lock and vice versa.
2. Display of lock and battery status information on the app.
3. Display of GPS location on the app.
4. Store and use a replaceable battery.
5. House a mechanical lock frame.

6. Perform electrical engagement/disengagement of a locking mechanism.

7. Be waterproof.

Not included in the project scope will be Bluetooth wireless connective capabilities, a fully autonomous locking mechanism (one that will both open/close and engage/disengage the lock wirelessly) and a rechargeable battery. We will also be ignoring extreme weather and temperature conditions and their impact on the material properties.

2.3 Characteristics of Intended Reader

The reader will have basic knowledge of how bikes are secured in public settings, namely using mechanical locks and keys or combination locks. They will understand why bike locking is necessary and what it means for a bike to be vulnerable to theft, i.e. that all main parts of the bike must be secured including the frame and wheels. They will also have basic knowledge of GPS and wireless communications that can be accessed through smartphone applications. Lastly, the reader will have a high-school level understanding of kinematic and electronics-related physics and math such that they can grasp the mechanical and electrical functionality of the system.

2.4 Organization of Document

The document will follow the base template provided by the Capstone 4TB6 course GitLab. It will be organized with an increasing order of specificity, beginning with the General System Description that will include an overview of the Smart Lock project. The Specific System Description will make up the bulk of the document, describing the project in greater detail with its assumptions, inputs and outputs, goals and models. Finally, since the context of the project is previously stated, the Functional and Non-Functional Requirements will be outlined, as well as any foreseen Likely and Unlikely Changes to the project. Lastly, the Traceability Matrix is included to define the relationships between various relevant formulas, sections and information in the document and how they be used to build the project.

3 General System Description

This section provides general information about the system. It identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints. [This text can likely be borrowed verbatim. —TPLT]

[The purpose of this section is to provide general information about the system so the specific requirements in the next section will be easier to understand. The general system description section is designed to be changeable independent of changes to the functional requirements documented in the specific system description. The general system description provides a context for a family of related models. The general description can stay the same, while specific details are changed between family members. —TPLT]

3.1 System Context

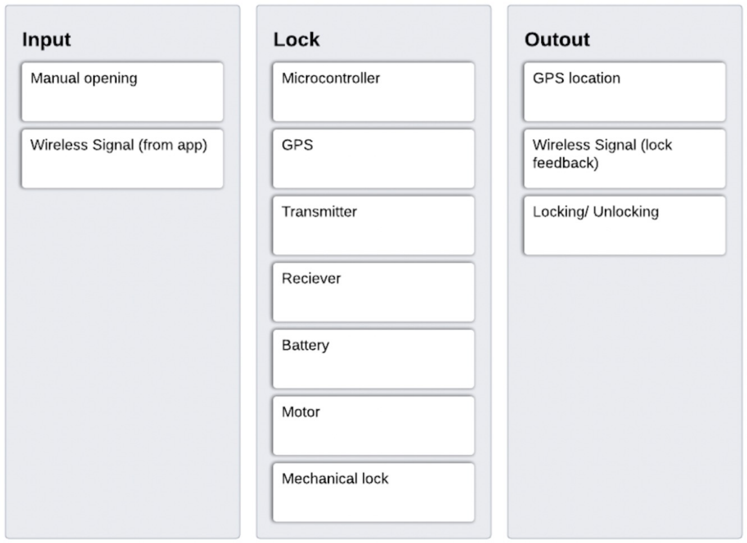

Figure 1: The System Context: Shows the inputs, the entities that manipulate them, and the outputs.

- User Responsibilities:
  -
- 4TB6 - Mechatronics Capstone Responsibilities:
  - Detect data type mismatch, such as a string of characters instead of a floating point number
  -

3.2 User Characteristics

[This section summarizes the knowledge/skills expected of the user. Measuring usability, which is often a required non-function requirement, requires knowledge of a typical user. As mentioned above, the user is a different role from the "intended reader," as given in Section 2.3. As in Section 2.3, the user characteristics should be specific an unambiguous. For instance, "The end user of 4TB6 - Mechatronics Capstone should have an understanding of undergraduate Level 1 Calculus and Physics." —TPLT]

3.3 System Constraints

The system has the following constraints.

- GPS to accurately locate the bike within a tolerance of 50 meters.
  - Justification: Complex software like Google Maps ensures an accuracy of around 20 meters. Considering users only require the general location and our software will not be as complex as Google's, the team added this constraint.
- Total mass of lock under 3 kg.
  - Justification: Average weight of a bike lock is just under 3 pounds. To ensure the lock was not significantly heavier the team added this constraint.
- App storage under 50 megabytes.
  - Justification: A small mobile app should not take significant space on the user's phone. Similar GPS applications are around 50 megabytes.
- The budget for testing, designing, and equipment is $750.
  - Justification: Smart Lock has a limited budget to work within to encourage efficient use of materials and ensure low costs

4 Specific System Description

4.1 Problem Description

There are many problems associated with bike locks today. People often forget or lose their keys, lock or combination. Additionally, current locking systems are often not comprehensive – they may not lock all parts of the bike that can be stolen, namely the seat, front and back wheels, and frame.

Furthermore, bike locks can be bulky, heavy and dangerous to carry around. It can also be tedious to find and lock one's bike to an external frame. The combination of these nuisances can lead to individuals leaving their bikes without properly securing them. The city of Toronto reports an average 3625 stolen bikes annually, and the Canadian Cycling Magazine estimates that only 15-20% of stolen bikes are reported, which indicates a rather expansive problem that can be solved [1,2].

4.1.1 Terminology and Definitions

The terms listed below will be used throughout the scope of this document and should be referenced when referring to specific components.

- Smart Lock: The scope of this project and document referring to both the mechanical lock and the mobile application communicating with it.
- Lock: The physical component of the Smart Lock; includes the hardware, battery, motor, and locking mechanism.
- Team: Refers to the members working on Smart Lock
- App: The mobile application used to communicate with the lock
- Open/Close: The process of physically moving the lock frame into a state where it can be latched onto something else.
- Open/Close Status: The current state of the lock frame.
- Engage/Disengage: The automated process to actuate the lock to secure the frame.
- Wireless Signalling: The means of communication between the lock and the mobile application.
- Battery: The power source used to power the motor for engaging and disengaging.
- Battery Status: A measure of the quantity of charge in the battery.
- Git: The platform used to store the documentation and software and track changes made to the files.
- XCode and Flutter: The IDE used to develop the mobile application.
- Cross-Platform Development: The means of developing the mobile application on both iOS and Android devices.
- Transmitter: The device used to send signals to and from the app to give status information and to send the engagement and disengagement signals.
- GPS: The satellite-based navigation system used to locate the bike.
- Receiver: The device used to receive the signals from the app for engagement and disengagement.
- Microcontroller: The programmable device used to control the motor.

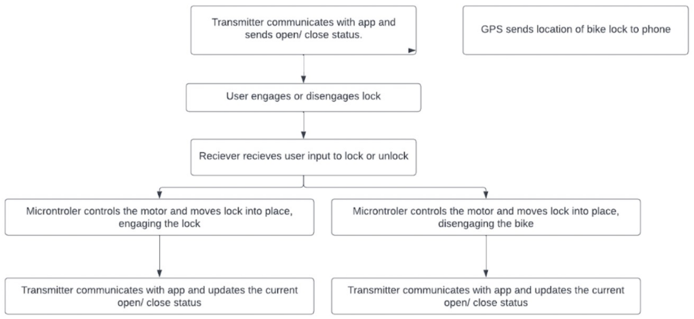
Transmitter communicates with app and sends open/ close status.
GPS sends location of bike lock to phone
User engages or disengages lock
Reciever recieves user input to lock or unlock
Microntroler controls the motor and moves lock into place, engaging the lock
Microntroler controls the motor and moves lock into place, disengaging the bike
Transmitter communicates with app and updates the current open/ close status
Transmitter communicates with app and updates the current open/ close status

Figure 2: The Physical System

4.1.2 Physical System Description

4.1.3 Goal Statements

Given the [inputs —TPLT], the goal statements are:

GS1: Effective Bike Lock: The lock is sturdy and cannot be manual opened by the average human once engaged.

GS2: Long Lasting Battery Life: The power source is used efficiently to last a sufficient period of time without requiring replacement.

GS3: Bicycle Versatility: The lock can be used on mountain, city, kids, and road bikes.

GS4: Easily Mountable on Bike Frame: Does not require special tools to be installed.

GS5: Effective GPS Tracking: The lock can accurately transmit location to app regardless of location.

GS6: Weatherproof: The lock is not easily damaged by wind, rain, snow, heat, and cold weather.

4.1.4 Assumptions

- The bike will be left in locations with strong GPS signal. Dependencies: Locations such as garages and underground parking.
- The user's phone will always be able to connect to the lock (wireless connection works as intended). Dependencies: All users have modern phones that can operate the application. The phones will also have functional location settings.
- Users will have no limiting physical disabilities. Dependencies: Average person is strong enough to manually lock and unlock the bike.

- The lock will be weatherproof. Dependencies: Rain, snow or extreme temperatures will not damage the lock.
- Battery life is only reduced when the motor is being used. Dependencies: Battery life is independent of weather and time.
- Efficiency is always constant. Dependencies: The battery's power does not decrease over time.

4.1.5 Theoretical Models

RefName: **Torque of a Motor**

Label: $\mathbf{T = F \cdot d = I \cdot k_T}$

Equation:

Description: The above equation gives the torque T (N m) required to actuate the opening and closing mechanism of the lock through the motor. Torque is equal to the force F (N) to open or close the lock, multiplied by the distance d (m) from the motor's axis of rotation. To extract current, torque is also equal to the load (motor) current I (A) multiplied by the motor's torque Constant k_T (N m A^{-1}).

The net force will be calculated once the design is finalized. The distance and torque constant will be dependent on the motor. These equations will be used to calculate the current required from the battery. They must be computed twice; once for engaging, and again for disengaging.

Notes: None.

Source: https://www.motioncontroltips.com/faq-whats-the-relationship-between-current-and-dc-motor

Ref. By: GD??

Preconditions for Torque of a Motor: None

Derivation for Torque of a Motor: Not Applicable

RefName: **Ohm's Law**

Label: $V = I \cdot R$

Equation:

Description: The above equation gives the voltage V (V), which is equal to the load (motor) current I (A) multiplied by the resistance R (Ω) of the motor. This can be used to find the rating of battery needed to power the motor. This calculation must also be done twice as outlined above.

Notes: None.

Source:

Ref. By: GD**??**

Preconditions for Ohm's Law: None

Derivation for Ohm's Law: Not Applicable

RefName: **Battery Life**

Label: $\mathbf{BL = \frac{BC}{I}}$

Equation:

Description: The above equation gives the battery life BL (h), which will be calculated by dividing the battery capacity BC (A h) selected and the load current I (A).

Notes: None.

Source:

Ref. By: GD??

Preconditions for Battery Life: None

Derivation for Battery Life: Not Applicable

4.1.6 General Definitions

[General Definitions (GDs) are a refinement of one or more TMs, and/or of other GDs. The GDs are less abstract than the TMs. Generally the reduction in abstraction is possible through invoking (using/referencing) Assumptions. For instance, the TM could be Newton's Law of Cooling stated abstracting. The GD could take the general law and apply it to get a 1D equation. —TPLT]

This section collects the laws and equations that will be used in building the instance models.

[Some projects may not have any content for this section, but the section heading should be kept. —TPLT] [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Number	GD1
Label	**Newton's law of cooling**
SI Units	$\mathrm{W\,m^{-2}}$
Equation	$q(t) = h\Delta T(t)$
Description	Newton's law of cooling describes convective cooling from a surface. The law is stated as: the rate of heat loss from a body is proportional to the difference in temperatures between the body and its surroundings. $q(t)$ is the thermal flux ($\mathrm{W\,m^{-2}}$). h is the heat transfer coefficient, assumed independent of T (A**??**) ($\mathrm{W\,m^{-2}\,{}^\circ C^{-1}}$). $\Delta T(t) = T(t) - T_{\text{env}}(t)$ is the time-dependent thermal gradient between the environment and the object (°C).
Source	Citation here
Ref. By	DD1, DD**??**

Detailed derivation of simplified rate of change of temperature

[This may be necessary when the necessary information does not fit in the description field. —TPLT] [Derivations are important for justifying a given GD. You want it to be clear where the equation came from. —TPLT]

4.1.7 Data Definitions

[The Data Definitions are definitions of symbols and equations that are given for the problem. They are not derived; they are simply used by other models. For instance, if a problem depends on density, there may be a data definition for the equation defining density. The DDs are given information that you can use in your other modules. —TPLT]

[All Data Definitions should be used (referenced) by at least one other model. —TPLT]

This section collects and defines all the data needed to build the instance models. The dimension of each quantity is also given. [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Number	DD1
Label	**Heat flux out of coil**
Symbol	q_C
SI Units	$\mathrm{W\,m^{-2}}$
Equation	$q_C(t) = h_C(T_C - T_W(t))$, over area A_C
Description	T_C is the temperature of the coil (°C). T_W is the temperature of the water (°C). The heat flux out of the coil, q_C ($\mathrm{W\,m^{-2}}$), is found by assuming that Newton's Law of Cooling applies (A**??**). This law (GD1) is used on the surface of the coil, which has area A_C ($\mathrm{m^2}$) and heat transfer coefficient h_C ($\mathrm{W\,m^{-2}\,°C^{-1}}$). This equation assumes that the temperature of the coil is constant over time (A**??**) and that it does not vary along the length of the coil (A**??**).
Sources	Citation here
Ref. By	IM1

4.1.8 Data Types

[This section is optional. In many scientific computing programs it isn't necessary, since the inputs and outpus are straightforward types, like reals, integers, and sequences of reals and integers. However, for some problems it is very helpful to capture the type information. —TPLT]

[The data types are not derived; they are simply stated and used by other models. —TPLT]

[All data types must be used by at least one of the models. —TPLT]

[For the mathematical notation for expressing types, the recommendation is to use the notation of **?**. —TPLT]

This section collects and defines all the data types needed to document the models. [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Type Name	Name for Type
Type Def	mathematical definition of the type
Description	description here
Sources	Citation here, if the type is borrowed from another source

4.1.9 Instance Models

[The motivation for this section is to reduce the problem defined in "Physical System Description" (Section 4.1.2) to one expressed in mathematical terms. The IMs are built by refining the TMs and/or GDs. This section should remain abstract. The SRS should specify the requirements without considering the implementation. —TPLT]

This section transforms the problem defined in Section 4.1 into one which is expressed in mathematical terms. It uses concrete symbols defined in Section 4.1.7 to replace the abstract symbols in the models identified in Sections 4.1.5 and 4.1.6.

The goals [reference your goals —TPLT] are solved by [reference your instance models —TPLT]. [other details, with cross-references where appropriate. —TPLT] [Modify the examples below for your problem, and add additional models as appropriate. —TPLT]

Number	IM1
Label	**Energy balance on water to find T_W**
Input	m_W, C_W, h_C, A_C, h_P, A_P, t_{final}, T_C, T_{init}, $T_P(t)$ from IM**??** The input is constrained so that $T_{\text{init}} \leq T_C$ (A**??**)
Output	$T_W(t)$, $0 \leq t \leq t_{\text{final}}$, such that $\frac{dT_W}{dt} = \frac{1}{\tau_W}[(T_C - T_W(t)) + \eta(T_P(t) - T_W(t))]$, $T_W(0) = T_P(0) = T_{\text{init}}$ (A**??**) and $T_P(t)$ from IM**??**
Description	T_W is the water temperature (°C). T_P is the PCM temperature (°C). T_C is the coil temperature (°C). $\tau_W = \frac{m_W C_W}{h_C A_C}$ is a constant (s). $\eta = \frac{h_P A_P}{h_C A_C}$ is a constant (dimensionless). The above equation applies as long as the water is in liquid form, $0 < T_W < 100^o$C, where 0^oC and 100^oC are the melting and boiling points of water, respectively (A**??**, A**??**).
Sources	Citation here
Ref. By	IM**??**

Derivation of ...

[The derivation shows how the IM is derived from the TMs/GDs. In cases where the derivation cannot be described under the Description field, it will be necessary to include this subsection. —TPLT]

4.1.10 Input Data Constraints

Table 1 shows the data constraints on the input output variables. The column for physical constraints gives the physical limitations on the range of values that can be taken by the variable. The column for software constraints restricts the range of inputs to reasonable values. The software constraints will be helpful in the design stage for picking suitable algorithms. The constraints are conservative, to give the user of the model the flexibility to experiment with unusual situations. The column of typical values is intended to provide a feel for a common scenario. The uncertainty column provides an estimate of the confidence with which the physical quantities can be measured. This information would be part of the input if one were performing an uncertainty quantification exercise.

The specification parameters in Table 1 are listed in Table 2.

Table 1: Input Variables

Var	Physical Constraints	Software Constraints	Typical Value	Uncertainty
L	$L > 0$	$L_{\text{min}} \leq L \leq L_{\text{max}}$	1.5 m	10%

(*) [you might need to add some notes or clarifications —TPLT]

Table 2: Specification Parameter Values

Var	Value
L_{min}	0.1 m

4.1.11 Properties of a Correct Solution

A correct solution must exhibit [fill in the details —TPLT]. [These properties are in addition to the stated requirements. There is no need to repeat the requirements here. These additional properties may not exist for every problem. Examples include conservation laws

Table 3: Output Variables

Var	**Physical Constraints**
T_W	$T_{\text{init}} \leq T_W \leq T_C$ (by A??)

(like conservation of energy or mass) and known constraints on outputs, which are usually summarized in tabular form. A sample table is shown in Table 3 —TPLT]

[This section is not for test cases or techniques for verification and validation. Those topics will be addressed in the Verification and Validation plan. —TPLT]

5 Requirements

[The requirements refine the goal statement. They will make heavy use of references to the instance models. —TPLT]

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

5.1 Functional Requirements

R1: [Requirements for the inputs that are supplied by the user. This information has to be explicit. —TPLT]

R2: [It isn't always required, but often echoing the inputs as part of the output is a good idea. —TPLT]

R3: [Calculation related requirements. —TPLT]

R4: [Verification related requirements. —TPLT]

R5: [Output related requirements. —TPLT]

[Every IM should map to at least one requirement, but not every requirement has to map to a corresponding IM. —TPLT]

5.2 Nonfunctional Requirements

[List your nonfunctional requirements. You may consider using a fit criterion to make them verifiable. —TPLT] [The goal is for the nonfunctional requirements to be unambiguous, abstract and verifiable. This isn't easy to show succinctly, so a good strategy may be to give a “high level” view of the requirement, but allow for the details to be covered in the Verification and Validation document. —TPLT] [An absolute requirement on a quality of the system is rarely needed. For instance, an accuracy of 0.0101 % is likely fine, even if the requirement is for 0.01 % accuracy. Therefore, the emphasis will often be more on describing now well the quality is achieved, through experimentation, and possibly theory, rather than meeting some bar that was defined a priori. —TPLT] [You do not need an entry for correctness in your NFRs. The purpose of the SRS is to record the requirements that need to be satisfied for correctness. Any statement of correctness would just be redundant. Rather than discuss correctness, you can characterize how far away from the correct (true) solution you are allowed to be. This is discussed under accuracy. —TPLT]

NFR1: **Accuracy** [Characterize the accuracy by giving the context/use for the software. Maybe something like, “The accuracy of the computed solutions should meet the level needed for <engineering or scientific application>. The level of accuracy achieved by 4TB6 - Mechatronics Capstone shall be described following the procedure given in Section X of the Verification and Validation Plan.” A link to the VnV plan would be a nice extra. —TPLT]

NFR2: **Usability** [Characterize the usability by giving the context/use for the software. You should likely reference the user characteristics section. The level of usability achieved by the software shall be described following the procedure given in Section X of the Verification and Validation Plan. A link to the VnV plan would be a nice extra. —TPLT]

NFR3: **Maintainability** [The effort required to make any of the likely changes listed for 4TB6 - Mechatronics Capstone should be less than FRACTION of the original development time. FRACTION is then a symbolic constant that can be defined at the end of the report. —TPLT]

NFR4: **Portability** [This NFR is easier to write than the others. The systems that 4TB6 - Mechatronics Capstone should run on should be listed here. When possible the specific versions of the potential operating environments should be given. To make the NFR verifiable a statement could be made that the tests from a given section of the VnV plan can be successfully run on all of the possible operating environments. —TPLT]

- Other NFRs that might be discussed include verifiability, understandability and reusability.

6 Likely Changes

LC1: [Give the likely changes, with a reference to the related assumption (aref), as appropriate. —TPLT]

7 Unlikely Changes

LC2: [Give the unlikely changes. The design can assume that the changes listed will not occur. —TPLT]

8 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 4 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 5 shows the dependencies of instance models, requirements, and data constraints on each other. Table 6 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

[You will have to modify these tables for your problem. —TPLT]

[The traceability matrix is not generally symmetric. If GD1 uses A1, that means that GD1's derivation or presentation requires invocation of A1. A1 does not use GD1. A1 is "used by" GD1. —TPLT]

[The traceability matrix is challenging to maintain manually. Please do your best. In the future tools (like Drasil) will make this much easier. —TPLT]

	T??	T??	T??	GD1	GD??	DD1	DD??	DD??	DD??	IM1	IM??	IM??	IM??
T??													
T??			X										
T??													
GD1													
GD??	X												
DD1				X									
DD??				X									
DD??													
DD??								X					
IM1					X	X	X				X		
IM??					X		X		X	X			X
IM??		X											
IM??		X	X				X	X	X		X		

Table 4: Traceability Matrix Showing the Connections Between Items of Different Sections

	IM1	IM??	IM??	IM??	4.1.10	R??	R??
IM1		X				X	X
IM??	X			X		X	X
IM??						X	X
IM??		X				X	X
R??							
R??						X	
R??					X		
R2	X	X				X	X
R??	X						
R??		X					
R??			X				
R??				X			
R4			X	X			
R??		X					
R??		X					

Table 5: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??
T??	X																		
T??																			
T??																			
GD1		X																	
GD??			X	X	X	X													
DD1							X	X	X										
DD??			X	X						X									
DD??																			
DD??																			
IM1											X	X		X	X	X			X
IM??												X	X			X	X	X	
IM??														X					X
IM??													X					X	
LC??				X															
LC??								X											
LC??									X										
LC??											X								
LC??												X							
LC??															X				

Table 6: Traceability Matrix Showing the Connections Between Assumptions and Other Items

The purpose of the traceability graphs is also to provide easy references on what has to be additionally modified if a certain component is changed. The arrows in the graphs represent dependencies. The component at the tail of an arrow is depended on by the component at the head of that arrow. Therefore, if a component is changed, the components that it points to should also be changed. Figure **??** shows the dependencies of theoretical models, general definitions, data definitions, instance models, likely changes, and assumptions on each other. Figure **??** shows the dependencies of instance models, requirements, and data constraints on each other.

9 Development Plan

[This section is optional. It is used to explain the plan for developing the software. In particular, this section gives a list of the order in which the requirements will be implemented. In the context of a course this is where you can indicate which requirements will be implemented as part of the course, and which will be "faked" as future work. This section can be organized as a prioritized list of requirements, or it could should the requirements that will be implemented for "phase 1", "phase 2", etc. —TPLT]

10 Values of Auxiliary Constants

[Show the values of the symbolic parameters introduced in the report. —TPLT]

[The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance. —TPLT]

[The value of FRACTION, for the Maintainability NFR would be given here. —TPLT]

[The following is not part of the template, just some things to consider when filing in the template. —TPLT]
[Grammar, flow and LaTeXadvice:

- For Mac users `*.DS_Store` should be in `.gitignore`
- LaTeX and formatting rules
  - Variables are italic, everything else not, includes subscripts (link to document)
    - Conventions
    - Watch out for implied multiplication
  - Use BibTeX
  - Use cross-referencing
- Grammar and writing rules
  - Acronyms expanded on first usage (not just in table of acronyms)
  - "In order to" should be "to"

—TPLT]
[Advice on using the template:

- Difference between physical and software constraints
- Properties of a correct solution means *additional* properties, not a restating of the requirements (may be "not applicable" for your problem). If you have a table of output constraints, then these are properties of a correct solution.
- Assumptions have to be invoked somewhere
- "Referenced by" implies that there is an explicit reference
- Think of traceability matrix, list of assumption invocations and list of reference by fields as automatically generatable
- If you say the format of the output (plot, table etc), then your requirement could be more abstract

—TPLT]

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.
2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?