BSSE SENIOR DESIGN PROJECT

HANDBOOK

VERSION 2.0

Faculty of Computing
Riphah International University

Table of Contents:

1. Glossary

SDP:	Senior Design Project (Also known as Final Year Project)
FYP	Final Year Project
BSSE:	Bachelor of Science in Software Engineering
FC:	Faculty of Computing, Riphah International University
RIPHAH:	Riphah International University
Faculty:	Fulltime permanent faculty member
Supervisor:	A fulltime faculty member of FC responsible for the supervision of SDP.
Examiner:	A professor or an expert of the relevant area chosen from outside FC, RIPHAH.
Coordinator:	A faculty member appointed by FC to coordinate the SDP tasks
Student:	A student registered for SDP in FC
Group/Team:	A group of students formed as a team to work on the SDP.

2. Introduction

The SDP is one of the most important aspects of BSSE degree program. It provides you with an opportunity to apply the principles learned in different courses, and integrate material learned at different stages of the program. There may also be a need for additional, domain specific knowledge in a project, which will necessitate additional study.

Learning Outcome: On successful completion of the project, you should be able to apply your knowledge and understanding of software engineering to analyzing problems, creating and evaluating solutions, and critically assessing your own work. You should also be able to prepare project plans, give presentations, and write reports.

Learning Method: The SDP is a self-learning exercise, under the guidance of project supervisor. The following six learning elements form the basis of a project: knowledge, understanding, applying, analyzing, evaluating, and creating.

During the project you are expected to:

- Develop an understanding of the technical knowledge on which your project is based.
- Analyze a problem.
- Create a solution (Synthesis).
- Apply your knowledge and understanding in executing the project.
- Evaluate your work.

Assessment

- Semester 1: Interim Report and Prototype
- Semester 2: Final Report, Presentation, and Demonstration

2.1. Focus

The objective of SDP is to develop a system that meets a set of user needs. This may take many forms, for example: an application, a server, a program, a library, a collection of programs, an embedded system, a hardware-software system, plug-ins, extensions, modifications to existing software etc.

You should focus on following sound software engineering principles, and evaluating your work thoroughly. You will do background research into both the technology and the marketplace. You must test your software to make sure it works correctly.

SDPs are based on the principles of quality software development, which can be summarized as follows:

1. Understand what is needed
2. Design a solution
3. Design and implement a system to realize the solution
4. Verify that the implementation matches the design
5. Validate that the implementation correctly realizes solution
6. Evaluate how well the solution solves the problem

The developed system and final report will be the main output of the project, which should show that your system is of suitable quality and solves user needs.

2.2. Importance of Senior Design Project

You will continue to gain experience after you graduate, but SDP will be your first exposure to a significant software engineering project. It is essential that you learn from this exposure, and practice all of the methodologies involved. It is also important that you learn not just how to apply what you know, but also to apply it with judgment, with the ability to assess what you are doing and to be critical of it.

You should keep in mind that:

- It is the largest single piece of work you will do during your degree course.
- It is the part of the curriculum that allows you to specialize in a topic you are good at or enjoy.
- It is the part of your degree program that prospective employers will most likely ask you about at an interview.
- It allows you to show off a wide range of skills and knowledge learned during your course.
- It encourages integration of material learned in a number of courses.

Responsibility: It is important to note that you are responsible for your project. Your supervisor will provide you with guidance and feedback, but you must put in the effort to achieve a successful project.

3. Selecting Your Project & Team Members

3.1. Project Team

One of the most important aspects of Software Engineering is team work; you will therefore need to form a group in order demonstrate your ability to engineer software as a team. Ideally a team of 3 members is encouraged; however a group of 2, 4 or 5 can be formed only if the scope and complexity of the project justifies these numbers.

Use "Template-01" to fill in the details of project team members and submit hard copy to Project Coordinator before the end of first week (1st Semester).

3.2. Choosing Your Project & Supervisor

Given that you are going to spend a lot of time working on your project, it is essential that you pick a project which you like and which you are capable of doing. Note that these are not necessarily the same things: just because you like a particular project doesn't mean you are qualified to do it. You may not have taken all of the requisite courses or it may be a more theoretically-aligned project whereas you might be a more practically-oriented student (or vice versa). Think long and hard before making your final choice. At the very least, you should take the following steps in assessing and choosing a project.

- Find out what are your options.
- Make a list consisting of at-least three projects.
- Write a brief descriptions of each project
- Go and talk to faculty members you wish to supervise you during the project. Discuss your ideas in detail and take notes. It is important that you demonstrate your interest and necessary knowledge about the projects in order to convince a faculty member of your choice to supervise you.

Once you have finalized your project & supervisor use "Template-02" to write down initial proposal and submit hard copy to project coordinator before the end of second week (1st Semester).

3.3. Finalizing Project Proposal

You will need to further refine your project proposal and present it to the entire faculty in fourth week of the semester (1st Semester). Faculty members will evaluate your proposal for **scope, complexity, originality of idea (creativity) and amount of self-learning involved**. Faculty members will provide feedback on your proposal, which you will need to incorporate in your project scope in consultation with your project supervisor. Faculty members can also reject a proposal if you fail to effectively present scope, complexity, originality of idea (creativity) and amount of self-learning involved.

Use "Template-03" to create project proposal presentation. You will need to submit soft copy (PDF Format) of your presentation to project coordinator before the end of fourth week (1st Semester).

After the proposal presentation you will need to finalize your project proposal (including initial plan) in consultation with project supervisor.

Use "Template-04" to create final project proposal and submit both hard copy (signed by project supervisor) and soft copy (PDF Format) to project coordinator before the end of sixth week (1st Semester).

Your final project proposal should consist of:

- Full Name & Designation of Project Supervisor
- Introduction: Briefly introduce users for which solution is being developed.
- Existing System/ Description of the Current Situation
- Problem Statement: Describe the motivation for new solution.
- Proposed Solution

- Scope of the Project: Discuss major modules in this section.
- List of Faculty Proposed Changes
- **Work Breakdown Structure:** A work breakdown structure (WBS) is deliverable based decomposition of project scope. The WBS includes 100% of the work defined by the project scope and captures all deliverables – in terms of the work to be completed, including project management.
- **Roles & Responsibility Matrix:** The purpose of roles & responsibility matrix is to identify who will do what. You will need to list items in WBS and identify activities needed to complete each deliverable. You will then identify which team member(s) will be responsible for completing which activity and how long it should take.

It is important to understand that at this stage you are only finalizing project proposal, therefore your project plan ("Work Breakdown Structure" and "Roles & Responsibility Matrix") can be based on rough estimates. You must regularly refine both "Work Breakdown Structure" and "Roles & Responsibility Matrix" as you elicit more requirements during your project.

4. Planning & Executing Your Project

One of the most important things you will learn when doing your project is the need to manage your time. Senior Design Projects are completely different from smaller projects you may have undertaken. They require a considerable amount of time: approximately more than 240 hours / person.

Begin your project early, work consistently at it, and track your progress. Your project plan helps you to think about all the things you need to do, their inter-relationships, and the time each will take.

4.1. Literature / Market Survey

Ideally you should conduct literature / market survey for problem identification. However even after you have defended your project proposal, you must conduct detailed literature / market survey to gain more insight about the problem you are trying to solve. This is an essential step that will help you become an expert in the problem at hand: a problem-domain expert.

- It shows that you not only understand what you want to do, but you understand what others have done related to your project. If you can tell people about what has already been done and what methods already exist, then your readers will think that at least you are interested in your topic, have some self-initiative and are informed and up-to-date.
- It shows that you are intelligent enough to evaluate the quality of the other work done on the subject, i.e., it shows that you are capable of thinking critically and identifying strengths and weakness.
- It gives you opportunity to tell how your project is related to previous work done by others on the subject.

- It tells the reader if you are simply going to duplicate others' work for the sake of gaining a better understanding, improve upon others' work or perhaps combine the methodology of two or more existing approaches to solving a problem.

The only way to become an expert in both the problem domain and the solution domain is to learn as much as possible about the area and to learn it as quickly and efficiently as possible.

Go online and search for articles, books and papers related to your subject. Be creative and persistent in your keyword search until you hunt down good references or examples.

- When you read some literature that you think is useful and related, first record the citation on your list of references, using IEEE format.
- When you read some literature that is not very useful, do not include it on your list of references. More references do not mean a better list of references. Useless references only confuse a careful reader and make you lose credibility.
- In each document, identify the approach(es) / method(s) for solving problem(s), and compare this/these with what you already know.
- Identify which approaches and/or methods you will use and omit in your project.

- After you feel satisfied that you know all or most of the existing approaches/methods, list out all the approaches/methods, in a logical sequence (perhaps in chronological order).
- For each approach/method: describe how it works and what its components are; mention its strengths and/or weaknesses; tell if you will use or omit the approach for your project and why or why not.

Finally, it is very important that you realize that, in order to fully understand anything that you read, you must write it up in your own words. If you can't express or speak about a given idea, then you haven't truly understood it in any useful way.

Based on your project proposal document and literature / market survey you should be able to complete <u>first two chapters</u> of your project report. Use "Template-05" to

create your project report and make sure there is an agreement on the contents between all team members and your supervisor.

In the first two chapter of your report you should cover:

- Chapter 1: Introduction
    - Background
    - Motivations and Challenges: Actual problem, why we feel to select this topic as our project
    - Goals and Objectives
    - Solution Overview: One or two page explanation about the solution which you wishes to recommend, approach will be on abstract level
    - Report Outline: Explaining which section will include what information, organization of the report will be explained here
- Chapter 2: Literature / Market Survey
    - Introduction
    - Literature Review/Technologies Overview: Existing models/ protocols/ industrial work, what sort of approach you will use.
    - Summary: The concluding remarks for the chapter and also motivation of next chapter

You should attach "**Work Breakdown Structure**" & "**Roles & Responsibilities Matrix**" as an appendix to the report.

4.2. Requirements Elicitation

Having chosen your project, you will have in your possession a brief description of what is involved in the project. You will realize by now that this is completely insufficient for you as a basis for doing the project. Consequently, your next task must be to find out exactly – and completely – what the project entails. This isn't as easy as it sounds. You might think that you should just ask your supervisor and he or she should tell you. It doesn't work like that. Quite often, a supervisor won't have an exact (and complete) model of what is required – supervisors are just like clients and customers in the business and industrial world. It is your job to help your supervisor identify exactly what he wants. That's what good software engineers do: they help people understand what they want and then they build it for them. Here's how you do it.

1. Talk to your supervisor.
2. Write down everything he or she says (by 'write down' I mean take notes of his or her words).
3. Write up everything he or she says (by 'write up' I mean express what your supervisor said in your own words).
4. Build a document describing what you think is required.
5. Go back to your supervisor and ask for his or her comments.
6. Return to step 1, and keep returning until you are both happy with your requirements document.

This all translates into one simple rule: find out what you want the final system to do and how it should behave, write it down, and get everyone involved to agree to it in writing. And don't spare the detail: every single aspect of what's wanted should be teased out and agreed: what it does, what it doesn't do, how the user is to use it or how it communicates with the user, what messages it displays, how it behaves when the user asks it to do something it expects, and especially how it behaves when the user asks it to do something it doesn't expect.

This process is called requirements elicitation because it reflects better the fact that you have to work actively with the client to find out what they really want (as opposed to what they initially say they want, which is a completely different thing). Once you have been though the requirements elicitation process several times and you are happy that you really know where you want to go, you must write it down and get everyone concerned to agree to it. This then becomes part of the system specification: it says what you are going to do.

4.3. Problem Modeling

Once you know the requirements, and are an expert in the problem domain, you can abstract the problem from the problem space and model it computationally: this means we can identify the theoretical tools we need to solve the problem. For example, if your problem is to do with building a database, you will probably model the system with an entity-relationship diagram and validate the model by normalization.

The key to all successful engineering is the use of an explicit model: if you don't have a model, you are probably not doing engineering. Connecting components (or lines of code) together is not engineering, irrespective of whether it works or not. Without the model you won't be able to analyze the system and, thereby, make firm statements about its robustness, operating parameters, and limitations.

4.4. Requirement Analysis & Specification

With the requirements document, problem definition and model identified, you can now say exactly what your system will do and under what circumstances it will do it.

Be careful not to involve design decisions in your analysis. Sometimes the design solution will fall out naturally from the analysis, but if you are making decisions then you have moved to the "Solution" activity.

In writing the specification, you should identify:

- Functional & Non-functional requirements
- The operational parameters (conditions under which your system will operate, including required software and hardware systems)
- Problem Scenarios: Failure modes and actions on failure
- Limitations & restrictions
- User interface or system interface

All this information is collectively known as the 'system specification' and is the result of an activity know as systems analysis.

Once you have this specification, before proceeding you must return and see if it actually matches what the user needs: i.e. you need to validate that the system specification satisfies the requirements (you would be surprised how often it doesn't). If it does, you can proceed to the next activity: software design. If it doesn't, then either the requirements were wrong and need to be changed, or the specification was wrong, and needs to be changed, or, more likely, both were wrong and need to be changed. You should get the explicit agreement of your supervisor that all is in order. If it isn't, then you must go back to requirements if necessary and revise them and the specifications (with your supervisor's agreement on everything). After this, you validate again, and you keep doing this until everyone agrees. Then, and only then, should you proceed to the next phase of the execution of your project: Design.

You should now be able to complete third chapter of your project report. Use "Template-05" to create your project report and make sure there is an agreement on the contents between all team members and your supervisor.

In the third chapter of your report you should at-least cover:

- Chapter 3: Requirement Analysis
    - Introduction: Explore or explain in detail the problem domain
    - Problem Scenarios: Critical scenarios in detail
    - Functional and Non functional Requirements

You should attach "**Software Requirements Specifications (SRS)**" as an appendix to the report.

4.5. System Design

You are now in a position to design your software system for implementation. This will show how the inputs are handled, how the processing is done, how internal data storage is achieved, and how the outputs are generated. At the software design stage you should not, for example, be making decisions about what the GUI will look like, but rather how you are going to implement it given the programming language/environment you are using.

Typically you need to use two levels of abstraction for your software design:

- **a high-level design**, showing the architecture of your solution (e.g. the major classes, and how they interact with each other)
- **a detailed design**, showing the details of the software components (e.g. the attributes and methods for each class).

You should now be able to complete <u>fourth chapter</u> of your project report. Use "Template-05" to create your project report and make sure there is an agreement on the contents between all team members and your supervisor.

In the fourth chapter of your report you should include relevant design specification (**do not reproduce whole of your design document, project report should give overview of the design and requirements**). Few important aspects of design that could be covered here are:

- Chapter 4: System Design
    - Architectural design representing all software/hardware components and the justification for using a particular architectural style/pattern.
    - Detailed design covering
        - Database/data design represented as ER diagrams/class diagram

- Appropriate UML diagrams representing the use case diagram, logical view, dynamic view and implementation view (only important aspects should be covered)
- Application of any design patterns
- In case of hardware related projects relevant circuit diagrams showing important components and their interactions.

You should attach "**Design Documents**" as an appendix to the report.

Use an incremental implementation approach. Don't attempt to build the entire system in one go in the hope that, when you switch it on or run it, it will work. This is the so called "Big Bang" approach (everything comes into existence at one instant) and its name is very appropriate, for it almost always results in initial chaos. It is much better to design, code, and test each software feature individually and add them to an ever expanding system. In general, it is best to complete the high-level design first, but then design, code, and test in an incremental manner.

4.6. Implementation

Finally, you are at the point where you can build your proposed system. Your implementation consists of writing the code and testing it. You will be expected to submit your source code for assessment. Make sure it is clearly structured, conforms to your coding standard, and is well documented and commented. **Make sure that any code that is not completely your own work is clearly identified**.

The implementation chapter (Chapter 5) in your project report should cover:

- Chapter 5: Implementation
    - Flow Control/Pseudo codes: Show the flow of your algorithm(s)
    - Provide list along with brief introduction to any Components, Libraries, Web Services or stubs that you are using in your system
    - Best Practices / Coding Standards that you are following.
    - Deployment Environment in which your system will be implemented
    - Summary

Complete fifth chapter of your project report. Use "Template-05" to create your project report and make sure there is an agreement on the contents between all team members and your supervisor.

4.7. Testing and Evaluation

Most students misunderstand the meaning of the word testing. They think it means showing that something works: their project, for example. But testing means much more than this. Certainly, you need to show that it works (i.e. that it meets the requirements and operates according to the specification), but a good testing strategy also attempts to break the system: to show not where it works but where it fails.

Simply writing "it is the best/fastest/most secure/most efficient etc" is meaningless. Statement like "tried our best, up hill task, dire need of time, indigenous effort, etc etc (stories)" must be avoided.

In engineering, we normally formalize the testing process by referring to three distinct goals:

- **Validation:** Simply stated, this test answers the questions: Have I built the right system? Does it satisfy the requirements? It may seem obvious, but you'd be surprised the number of times that the system which is built isn't what is wanted at all. You should compare the system's behavior with the original requirements and system specification. Validation is extremely important and it should be carried out with great attention to detail.
- **Verification:** In this case, the questions are: Have I built the system right? Is it computing the right answer? This is what most people understand by testing.
- **Evaluation**: Finally, we ask: How good is the system? Again, the hallmark of good engineering: we seek to assess the systems performance and compare it to that of other similar systems. Ideally, you should identify some quantitative metric by which to compare the systems, since numbers are the best and perhaps the only way to objectively describe performance.

The testing and evaluation chapter (Chapter 6) in your project report should cover:

- Chapter 6: Testing and Evaluation

- Introduction: Describe which aspects are you focusing on when testing software and hardware environment.
- List of Test Scenarios
- Performance and Evaluation: Consist of results and comparisons
- Summary

Complete sixth chapter of your project report. Use "Template-05" to create your project report and make sure there is an agreement on the contents between all team members and your supervisor.

You should attach "**Test Scenarios**" as an appendix to the report.

5. Documentation

Writing is an essential part of understanding. In case of documentation, writing is essential in order for others to understand what you have done. There are two reasons why you want others to understand your work:

- So that you can be given credit for it (your marks depends on it);
- So that others can carry on your work and develop or maintain your system.

It is extremely important that you document your work at every stage of your project. We saw already that documentation is essential in the initial reading-in, requirements, and specification phases but it is equally important in the design, implementation, test, and maintenance phases.

The best way to organize your writing is to keep a log book of all work in progress. You should go out and buy a nice hard-cover notebook and write everything you do on the project into this log book every day. Every thought and observation you have on your project should go into this book, along with notes of meetings with your supervisor, results, theoretical developments, calculations, everything. This log book will become an invaluable source of material when you come to write up your project in the final report.

5.1. Project Report

Your project report is a critical part of your project. It defines what you have done and why you have done it. It is also one of the main outputs on which your project will be examined and graded.

Your final year project provides you with an opportunity to demonstrate your ability to use your judgment. This means that you must show your skill at **assimilating** (to incorporate, understand), **synthesizing** (to produce, blend) and **critically appraising** (to critically evaluate) all material relevant to the project.

Your main opportunity to display your talents at assimilation and synthesis comes when you describe the background material you read, the requirements, specification, and design phases of your work. Needless to say, synthesis means that you must write the text yourself, expressing your understanding of the material in your own words.

Resist the temptation, no matter how strong, to copy sentences or paragraphs (or whole sections) from other books or articles. Copying is not synthesis and it demonstrates neither your assimilation of material nor your understanding of it. If you do come across a sentence or paragraph which is so good that is just has to be used, then do so and include it as a direct quotation, providing a reference to the original source.

It is extremely important that you also appraise, or assess, your work critically, i.e. with objectivity and with a view to seeing how it could be improved. Such honest criticism does not mean you will receive fewer marks; in fact, it is likely that you will receive more. Typically, you exercise your talents of critical appraisal at the end of the report in "conclusion and outlook" chapter (Chapter 7) but you can also exercise it throughout the report wherever it seems appropriate. Note that this exercise of critical appraisal is different from the testing processes of verification, validation, and evaluation, which refer to the functionality of the system you have designed. The critique you write applies less to the system and more to the overall objectives, methodologies, and findings of the project.

The conclusion and outlook chapter (Chapter 7) in your project report should cover:

- Chapter 7: Conclusion and Outlook
    - Introduction
    - Achievements and Improvements
    - Critical Review
    - Future Recommendations/Outlook: How much work we have done and in what directions we can do the future work
    - Summary

Complete seventh chapter of your project report. Use "Template-05" to create your project report and make sure there is an agreement on the contents between all team members and your supervisor.

The structure of your report is critical to the impact you make in your writing. Remember that you are trying to convey a message to the reader and, since this message is likely to be quite complicated, you must assist him or her by making your points clearly and in a logical order.

Because all engineering projects involve (or should involve) the exercise of a fairly standard engineering methodology (requirements, specification, modeling, realization, testing & evaluation), you can, if you wish, adopt a fairly standard structure. However, that this does not mean that you just simply have to fill in the gaps in a general report template (**Template-05**): this standard structure simply provides you with a place to start as you begin to design the final structure and content of your report. You will still have to do quite a lot of work to make it fit your own project.

5.2. A Very Short Guide to Good Writing

Good writing is difficult. It takes practice and a willingness to revise your work, many times. The goal of writing is to convey a message to the reader. However, writing, and reading, are sequential processes. Therefore, you have to construct the meaning of your message, piece by piece, in a linear time-line. However, the meaning you intend to convey may emerge from many sources, not all related in a nice orderly fashion. This creates a problem for the writer: how to order the messages contained in each sentence effectively. The job of a writer is to make the sequence of pieces as mutually-relevant as possible so that the story or message builds naturally, each piece adding to the previous one. When the pieces are presented out of order the impact on the reader is lessened. And remember the golden rule in writing: keep it simple.

The following are some pointers to help you in your task:

- Use short sentences and make sure the sentences are complete.
- Good writing strikes a balance between short sentences and longer more descriptive ones. Just as in oral communication, the full stops mean pauses: too many pauses and the text sounds disconnected, too few and it can be hard to follow the story line. Strike a balance but favour brevity over complexity.
- If you use pictures and diagrams (and you should), make sure each one has a self contained explanatory caption. Never refer to a picture or diagram in the main text without saying what it is. For example, never say "Figure 2.3 shows the results of the noise test" and then carry on to another topic. Help the reader. Summarize the content of the figure in a short sentence: "Figure 2.3 shows the results of the noise test which demonstrate the robustness of the system to Gaussian noise with a standard deviation of 2.3 or less." If you have copied the figure from a book or article you must cite the source.
- Make the paragraph your unit of construction. Each paragraph should bind one or more sentences about a given subject or idea. If the subject or idea changes, start a new paragraph.

- Omit needless words. Unnecessary words distract the reader. Don't write, "This is a system the performance of which is very useful". Instead, write, "This is a useful system".
- Write in a way that comes naturally. Speak the sentence. If it sounds correct, trust your ear and use the sentence. If it sounds unnatural, rewrite it.
- Avoid fancy words; they don't impress anyone.
- Be clear in your expression. If the idea you are trying to convey is getting lost in a sea of words and phrases, draw a line through the sentence and start again.
- Don't take short-cuts. Explain what you mean. Don't leave the reader to struggle trying to figure out what is the real meaning of your carefully constructed but concentrated sentence. He may conclude there is none. Explain all acronyms the first time you use them.
- Revise and rewrite. It is highly unlikely that you will manage to find the best way of expressing an idea with your first attempt. Nonetheless, make that attempt and then be prepared to revise it, again and again.

Remember, if you want to learn how to write a good report, you need to do two things: you need to read other good reports and you need to practice your own writing.

5.3. Giving Presentations

During the course of your project, you will be required to give three presentations:

1. Proposal presentation to entire faculty in 4th week of your first semester. (**Use Template-03**)
2. Progress presentation with your prototype demo to entire faculty in week immediately after finals of your first semester.
3. Final presentation along with your internal demo to entire faculty in 10th week of second semester. (**Use Template-06**)
4. Final presentation to external examiner anytime between 14th and 16th weeks of second semester. (**Use Template-06**)

You have learned much about presentation skills during your degree program and it wouldn't be appropriate to attempt to review everything you have been taught already. However, a few pointers may help you give a professional and impressive presentation:

- Don't depend too much on PowerPoint slides: your speech is the presentation and the slides support you (not the other way around).
- Take your time: pause frequently. Sometimes, the best thing to say is nothing. Short one-second rests create dramatic impact and also give your audience time to assimilate what you've said. Of course, you also have to maintain continuity and flow; otherwise people forget what you are talking about. It's a question of balance.
- Arrive early and make sure you know where all the equipment is. Know how to use it.
- Look at the audience, not at your slides.
- Project your voice (but don't shout).
- Smile: enjoy giving your presentation.
- Be confident: you've done some great work - here is your opportunity to get credit for it.
- The people in the audience are on your side (though sometimes they disguise it well!) They want you to succeed. If they ask you a question you don't understand, say so and ask their help. Ask them to explain, and ask nicely. If you still don't understand, don't bluff. Admit your ignorance and suggest ways of how you will overcome that lack of knowledge.
- Nobody knows everything; but that's no excuse for not trying to know everything. A knowledgeable person knows enough to do his job well, a wise person knows that he doesn't know everything, and an intelligent person knows how to find out what he doesn't know. Be knowledgeable, wise, and intelligent: be an engineer.

6. Project Assessment

6.1. Senior Design Project (Part I)

Your supervisor will grade you based on your project report and system prototype. You are required to complete first five chapter of the report and make enough progress in terms of actual implementation to call it a prototype.

- 30% Marks: Prototype
- 50% Marks: Project Report (Chapter 01 – Chapter 05)
- 20% Marks: Any other criteria set by supervisor

You will be required to participate in the **Open House (14th Week of first semester)** and showcase your Project Idea, Prototype, and Project Report. Failure to participate will earn you **"F" grade** unless you have prior approval from both your supervisor and project coordinator.

You will also be required to make a **progress presentation** with your report & prototype demo to entire faculty in week immediately after finals of your first semester.

6.2. Senior Design Project (Part II)

An external examiner (someone from outside the university – usually someone from another university or industry) will grade you based on your final report, presentation, and demonstration.

You will be assessed by entire faculty in an **Internal Demo (10th Week of second semester)**. Faculty members will decide whether you have successfully completed your project to qualify for the external exam or not based upon difficulty level (scope,

complexity, creativity and amount of self-learning involved in the developing the final system) and quality (presentation, usability, system design, documentation, and coding).

- In case they decide against you, you will earn an **"F" grade**.
- In case they decide in your favor, they will make a second decision regarding quality of your project. If your project is of good quality, faculty will allow you to participate in the **Open House (14th Week of Semester 2)**. External examiners are aware of this decision which may affect your grade in the second semester.

In case faculty has voted you an "F" grade, you may file an appeal and ask for re-evaluation.

Internal Demo (Part II) Appeal & Re-Evaluation Terms:

- Project groups with at-least 30% positive votes qualify to file an appeal against internal demo results.
- Request for re-evaluation should be submitted to project coordinator within 7 days of the Internal Demo.
- Project will be re-evaluated within 10 days of the appeal letter.
- Project will be re-evaluated by committee of 3 or more faculty members appointed by In-charge undergraduate program in consultation with dean.
- Decision of this committee will be final.

7. Schedule & Milestones

7.1. Senior Design Project (Part I)

Deadline	Milestone	Submit To	Notes / Actions / Deliverables
Week 1	Attend Orientation Seminar.	Project Coordinator	• Submit Project Team List (HARD COPY - Use Template-01)
Week 2	Develop Project Ideas; Find a Supervisor	Project Coordinator	• Submit Initial Proposal (HARD COPY - Use Template-02)
Week 4	Defend Proposal	Entire Faculty	• Project Proposal Presentation (SOFT COPY - Use Template-03)
Week 6	Finalize Project Proposal	Project Coordinator	• Submit Project Proposal & Plan (Both SOFT & HARD COPY - Use Template-04)
/Week 14	Open House	Industry, Faculty & Students	• Banners, Posters, Brochure, Story Board, Prototype, Project Report
Week Immediately After Finals	Progress Presentation / Assessment	Entire Faculty / Supervisor	• Prototype, Project Report (SOFT COPY - First 5 Chapters – Use Template-05)

7.2. Senior Design Project (Part II)

Deadline	Milestone	Submit To	Notes / Actions / Deliverables
Week 9	Submit Report	Project Coordinator	• Complete Project Report (SOFT COPY - Use Template-05)
Week 10	Internal Demo	Entire Faculty	• Final Presentation (Use Template-06) • Full Working Demo
Week 14	External Demo / Open House	External Examiner / Industry, Faculty & Students	• Banners, Posters, Brochure, Project Report • Final Presentation (Use Template-06) • Full Working Demo
Week Immediately After Finals	Submit Project	Project Coordinator	• Documentation (Hard Binding – 3 Copies) • CD (Including Software – Source Code, Appendix in PDF Format)

8. Appendices

- Template-01: Project Team
- Template-02: Initial Proposal
- Template-03: Proposal Presentation
- Template-04: Proposal & Plan
- Template-05: Project Report
- Template-06: Final Presentation

Template-01: Project Team

Microsoft Word Format

Template-02: Initial Proposal

Microsoft Word Format

Template-03: Proposal Presentation

Microsoft PowerPoint Format

Template-04: Proposal & Plan

Microsoft Word Format

Template-05: Project Report

Microsoft Word Format

Template-06: Final Presentation

Microsoft PowerPoint Format