SE+CS Capstone Project Status Sheet

This form attempts to give a broad project overview. It is about breadth, not depth. For any given project, there will be parts of this form that are more and less important.

Your presentation is where you will go in depth on the things that are of particular interest for your project. This form is to identify interesting issues, but does not have space to explain them thoroughly.

Contents

1 *Project & Team Identification*

TeamName : Project

Abstract: See Handbook for instructions.

Students

1.1 *Opportunity Assessment*

What do you want to learn?

What are you hoping to learn?

What you want to learn might include Technical Elective courses that complement your project. You might be learning the content of those courses through capstone rather than by taking the course.

What is the opportunity in the world?

Why does the world need your project? Why is now the right time to do it?

Why does the world need your project? Why is now the right time to do it? Has something changed in the world to create this opportunity now? Is this a longstanding opportunity?

Advantages		*Challenges*	
External Partner	☐	*Access to Data*	☐
New Technology	☐	*Against Vested Interests*	☐
New Data Source	☐	*Dense Market Space*	☐
Clear Opportunity/Prob.	☐	*Significant Marketing*	☐
Clear User Population	☐	*Scope Too Large*	☐
Domain Knowledge	☐	*Scope Too Small*	☐
Novel Idea	☐	*Scope Too Foggy*	☐
Grounded Idea	☐	*Free Engineering Labour*	☐
Awesomeness	☐	*Legal/Regulatory Hurdles*	☐
Other	☐	*Other*	☐

These checklists have some common advantages and challenges that have been faced by projects in the past.

Free or undervalued engineering labour is a strategic blunder if it is the basis of a proposed commercial competitive advantage: *i.e.*, we can undercut the competition because they pay their engineers and we do not value our own work. This violates P.E.O. Code of Ethics 77(7)v. Note that *pro bono* work in the public interest, including patches to established FOSS projects, does not suffer this blunder.

Summary of Advantages

Summary of advantages.

Strategies for Mitigating Challenges

Strategies for mitigating identified challenges.

1.2 Curriculum Connections

Course	*Who?*	*Why?*
SE463	everyone	project needs a spec
SE464	everyone	project needs a design
SE465	everyone	project needs testing

Identify courses with some connection to your project. It might be the case that your project goes beyond what is actually taught in the course — list the course here anyways.

It's ok if nobody on your team is taking the course. Capstone gives you a change to learn ideas from other courses on your own. Still list the course here, as there is a connection.

CRUD projects need to list CS349 user interfaces and CS449 / MSCI343 / SYDE548 user-centred design. CRUD projects often have no algorithmic technical depth: their technical depth comes from human-computer interaction / user-centred design.

CRUD is an old industry acryonym for Create/Read/Update/Delete. In the modern context, a database-backed website or app. CRUD is a very popular technical category.

2 *Applying the Wisdom of the Past*

2.1 *Software Engineering Rules of Thumb*

Lehman Type:

Discuss Lehman Type of your project

Perlis Epigram:

Discuss application of Perlis epigram to your project

Another Law:

Discuss application of another law / rule of thumb to your project

2.2 *Learning from Prior Capstone Projects*

Project 1:

Discuss what you learned from a past capstone project 1

Project 2:

Discuss what you learned from a past capstone project 2

Project 3:

Discuss what you learned from a past capstone project 3

3 *Feedback Given and Received*

3.1 *Responses to Feedback Received*

Party	*Date*	*Focus*	*Outcome*
▾		▾	▾
▾		▾	▾
▾		▾	▾
▾		▾	▾
▾		▾	▾

3.2 *Summary of Feedback Given*

Team	*Date*	*Comment*

4 Processes, Practices, and Tools

4.1 Tools

Tools you used for version control, testing, prototyping, *etc.*

4.2 Process

Briefly describe your team's process.

Briefly describe your team's process

Process Alternatives

Briefly describe some alternative processes you have used on co-op work terms and why your team isn't using them.

Describe allternative proceesses from your co-op work terms and why your team isn't using them.

4.3 *Process Maturity Assessment*

Joel Test:	
CMMI Capability:	
CMMI Maturity:	
UX Maturity:	
Other:	

Describe your team's process.

There are several process assessments in Software Engineering, including those listed here. Perhaps you know of another interesting one.

The process assessments listed here often apply to entire organizations, and not just individual projects, but nevertheless many aspects are applicable to individual projects.

The Joel Test was defined by Joel Spolsky as a lightweight assessment.

CMMI Capability and Maturity levels are defined by the Software Engineering Institute at Carnegie Melon University (CMU). For individual project teams it only makes sense to consider CMMI levels 0–2.

UX Maturity levels are defined by the Nielsen/Norman Group.

Discussion of Process Maturity

4.4 *Additional Practices*

Brief discussion of your team's professional practices of interest, such as code review, pair programming, *etc.*.

5 *Requirements & Specification*

5.1 *Exploratory Activity Summary*

Figma is one example of a tool for creating *Paper/LowFi prototypes*. §10 and §11 describe dozens of possible user activities. §5.4 describes different kinds of prototypes, including experimental prototypes. *Experimental prototypes* are ones that you build to learn knowledge, and then the prototype is discarded.

Client Meetings	0	*Exploratory User Activities*	0
Paper/LowFi Prototypes	0	*Experimental Prototypes*	0
Formal Logic Models	0	*Research Literature Report*	0

Description of exploratory activities.

5.2 *SRS Summary*

Relevant means that this criteria is relevant for your project. For example, computational complexity is not relevant for most CRUD projects. Security might not be relevant for a project to paralellize a complex computation. And so on.

Specified means that you have written out a specification for this criteria.

Satisified means that you have an implementation that satisifies the specification (and some evidence to support that claim). Design, implementation, deployment, verification, validation, *etc.*, are discussed more fully in subsequent sections.

	Relevant	*Specified*	*Satisfied*
Opportunity	☐	☐	☐
Functionality	☐	☐	☐
Correctness	☐	☐	☐
Computational Complexity	☐	☐	☐
Performance	☐	☐	☐
Privacy	☐	☐	☐
Security	☐	☐	☐
Usability	☐	☐	☐
Dependability/Availability	☐	☐	☐

5.3 *Deviations from SRS*

Explain the deviations in more depth in your presentation. This box is just to identify the issues.

Identify Deviations from SRS.

6 Design, Implementation, and Deployment

6.1 Applied Foundations

Identify important or interesting foundations that you applied in your project.

Algorithms:	
Data Structures:	
Math & Science:	
Standards:	

6.2 High-Level Design

Anticipated Variabilities are things in the project domain that are expected to change in the future. Design patterns are often ways of designing for anticipated future changes.

Architectural Style(s):	
Architecture Rationale:	
Anticipated Variabilities:	
Key Design Decisions:	

6.3 Components/Libraries

Component/Library	*Normal*	*Rationale / Comment*
	☐	
	☐	
	☐	
	☐	
	☐	
	☐	

Normal means used as the component provider intended, or as is commonly done. Most designs use normal components in normal ways.

The contrast to *normal* is *radical* or *novel* or *innovative*. This is interesting (and risky), so worthy of further discussion (in your presentation).

6.4 Implementation

Languages Used:	
Technical Debt(s):	

Development		*Lines Of Code*	
· *Commits*	0	· *Prototype(s)*	0
· *Merge Requests*	0	· *Experiments*	0
· *Issues Opened*	0	· *Build scripts etc.*	0
· *Issues Closed*	0	· *Tests (LOC)*	0

Lines of code should be measured with CLOC: `https://www.tecmint.com/cloc-count-lines-of-code-in-linux/`
sudo apt install cloc
LOC should not be measured by Git.

report from cloc

6.5 Deployment

Containerization:	e.g. Docker
Cloud Services:	e.g. rsync.net
Environments:	e.g. dev/test/prod or blue/green etc.

7 *Verification & Validation*

7.1 *Testing*

Coverage metrics include input space, line, branch, path, UI, *etc.*

Testing Strategies	
Unit Tests	0
Integration Tests	0
System Tests	0
UI Tests	0
Fuzz Testing	0
Random Testing	0
Generated Tests	0
Invariants	0
Alternative Implementations	0
Inverse Functions	0
Statistical Cross-Validation	0
Continuous Integration	0
Other	0

Testing Coverage

Coverage metrics and measurements.

7.2 *Summative User Activities*

Summative user activities

Briefly describe any summative user activities you have done to evaluate how well your project meets user needs. Your presentation will have a more complete description of these activites (if relevant). Here you can cover the basics, such as:

- ☐ question(s) being investigated
- ☐ kind of activity
- ☐ date(s) of activity
- ☐ *n* (number of participants)
- ☐ time per participant

7.3 *Results Target(s)*

Please also indicate progress towards the targets.

Rubric:	
Real World:	
Pitches:	
Competitions:	
Awards:	
Other:	

8 *Graduate Attributes Self Assessment*

See the *Learning Objectives* chapter of the handbook for a listing all of the *capstone relevant* graduate attributes in each category. This table also lists the *total* number of graduate attributes in each category.

Your team will self-assess how many of these graduate attributes are relevant to your project, and of those *project relevant* attributes, how many your team has *achieved*.

Category	*Achieved*	*Project Relevant*	*Capstone Relevant*	*Total*
Knowledge Base	0	0	3	7
Investigation	0	0	3	3
Problem Analysis	0	0	10	10
Specification	0	0	3	3
Design	0	0	8	8
Programming Technology	0	0	6	6
Implementation	0	0	4	7
Verification & Validation	0	0	9	12
Maintenance	0	0	3	4
Individual Work	0	0	5	6
Team Work	0	0	4	4
Communication Skills	0	0	5	5
Economics	0	0	5	6
Project Management	0	0	4	6
Tools	0	0	5	5
Professionalism	0	0	7	7
Impact of Engineering on Society and the Environment	0	0	3	5
Ethics and Equity	0	0	3	3
Lifelong Learning	0	0	4	4

Highlight Graduate Attributes

Highlight graduate attributes

List graduate attributes that are particularly strong in your project

TBD/Irrelevant/Omitted Graduate Attributes

Highlight graduate attributes

List graduate attributes that are TBD (To Be Done), irrelevant, or omitted from your project