UMBC

Capstone in Software Engineering Handbook
SENG 701

UMBC

Revision History

Date	Version	Notes
Feb. 10, 2023	v0.1	Initial Capstone project write-up
June 10, 2023	v0.2	Compiling multiple sections together
July 10, 2023	v0.3	Research and review of other Capstone courses at UMBC and external universities
July 27, 2023	Draft 1	Initial handbook with complete outline
August 6, 2023	Version 1	Version 1 text with all sections
August 29, 2023	Version 1.1	Updated checkpoints text, full proofread, minor edits and formatting, appendix citations, spell and grammar recheck
August 30, 2023	Version 1.2	Final version for Fall 2023
January 29, 2024	Version 1.3	Updated sponsor criteria and capstone proposal template
August 26, 2024	Version 1.4	Added additional text to clearly state that an e-commerce application or website are not acceptable capstone projects
January 26, 2025	Version 1.5	Removed RC checkpoint requirements. Clarified deliverables for checkpoints.
August 27, 2025	Version 1.6	Added Appendix 8 with submission checklists. Clarified user stories and lo-fi or hi-fi prototype.
August 29, 2025	Version 1.6.1	Added Git repository requirement to the Proposal Deliverable Checklist (Appendix 8) and all Capstone deliverables.

Introduction

The Capstone in Software Engineering course provides an opportunity for students to carry out an individualized significant project of supervised research or applied project in a specific area in software engineering. The deliverables from this course can be in the form of a research paper ready for peer review publications, a new or improved software app, application or information system, a consumer facing device, a solution for a non-profit community partner, a novel software engineering technique, a software engineering tool, a business and technology plan for a new startup, or a minimally viable product, or a full functional prototype of a software intensive product. The deliverables should make an original contribution to the software engineering body of knowledge or the software engineering professional practice and demonstrate advanced skills in at least one software engineering discipline.

Software Engineering MPS students who would like to enroll in this course need permission from the course instructor, department chair, and must have completed 18 graduate credits. Graduate students outside of the Software Engineering MPS program who are interested in enrolling in this course must get permission from their academic advisor and department chair, must have completed at least 18 credits in their graduate program with a GPA of 3.2 or higher, and need to request permission via this form: https://bit.ly/seng-courses.

This handbook was developed based on research of internal UMBC programs and external programs at other universities. We are grateful for resources we received from other UMBC programs to assist in developing this handbook particularly Biotechnology, Cyber Security, Data Science, HIT, and I/O Psychology. Our research also included Florida Tech, Georgia Tech, University of Florida, and Virginia Tech.

Purpose

The purpose of the capstone is to demonstrate an advanced level expertise in building software and software intensive products, services, and tools including the following:

- Managing, implementing, and executing a full life-cycle software engineering project and working with end-users, the community, and industry partners to plan a software engineering project.
- The ability to create a fully functional prototype or a minimally viable software intensive product (MVP).
- The ability to visualize product benefits, features, and future plans into a compelling story.
- The ability to communicate and present effectively to a wide range of audiences including end-users, people from other fields, and non-technical businesspeople.
- The ability to communicate in writing through a detailed report and summaries for different audiences from an executive summary to a detailed technical specification.

Requirements

The Capstone project must meet the following requirements:

1. **Building a Software Product**: The Capstone project must include designing, building, and developing a solution using software and software intensive products to solve or improve a significant real-world problem, issue, gap, or question.

2. **Sponsor**: The Capstone project is conducted to assist a business organization, a non-profit entity, or a community group. The sponsor organization business must be in agreement and alignment with the university mission and vision with a peaceful and purposeful mission. Certain limitations and exclusions apply not limited to no weapon making machines or systems, gambling, addictive applications, applications with sexual content or racial content. The sponsor must meet the criteria in Appendix 7.

3. **Relevant Problem**: The Capstone project must address a real-world problem, issue, gap, or question that is relevant to modern living and working impacting living and working conditions faced by individuals, teams, groups, and organizations and cannot be about a theoretical way of doing things.

4. **Achievable Solution**: The Capstone project requires a problem, issue, gap, or question that can be addressed through the application of software engineering principles, methods, and techniques by understanding and improving the living conditions, performance, and productivity of individuals and organizations.

5. **Independent New Project**: The Capstone project must be an independent work effort. The student will conceptualize the project in collaboration with a client to address an immediate business or community need. The work performed must be sufficiently distinct and different from the student's current or past job tasks, responsibilities, or duties.

6. **Software Engineering Methods**: The methods and techniques used in building the software or the software intensive product must have at least five techniques from three or more of software engineering subfields: Software Requirements, Software Design, Software Construction, Software Testing, Software Maintenance, Software Configuration Management, Software Engineering Management, Software Engineering Process, Software Engineering Models and Methods, and Software Quality.

7. **Software Process Management**: The Capstone project must be designed and built using an appropriate software process management based on rich or agile process models. It must adhere to the phases required to complete the project including a proposal, alpha, beta and final. Adherence to an appropriate software process also includes utilizing version control systems like Git/GitHub with frequent commits to document progress and maintain a complete and undeletable history of development efforts.

8. **Requirements Engineering**: The product requirements must be obtained through Requirements Engineering techniques including but not limited to literature review and multi-level analysis including best practices, market, trends, and competitive analysis, direct data collection through interviews, surveys, and focus groups, indirect data collection through on-site individual and group observation, document analysis, and workflow documentation. The requirements aim to provide a diagnosis and holistic analysis based on multiple perspectives in order to determine the root causes of the problem, issue, gap, or question.

9. **Measurable Objectives**: The student will identify five main project objectives. The objectives must be measurable and trackable. The success of the project is assessed based on the achievement of these objectives.

10. **Graduate Level Work**: The Capstone project must exhibit a level-of effort, problem complexity, and product output and product quality appropriate for a graduate level Capstone project. It is expected that the Capstone project will require a minimum of 150 hours of effort with efforts of at least 50 hours per month over three months. See Appendix 1 – Acceptable and Non-Acceptable Capstone Project Examples.

Textbook, Readings, and Resources

The Capstone does not have a required textbook. The following are recommended readings and resources:

- Designing Software Intensive Products, Rogerio Atem de Carvalho
- Engineering Software Products: An Introduction to Modern Software Engineering, Ian Sommerville
- Art of the Start 2.0: The Time-Tested, Battle-Hardened Guide for Anyone Starting Anything, Guy Kawasaki
- SEI Software Engineering: Building Systems from Commercial Components, Kurt Wallnau, Robert Seacord, Scott A Hissam
- Software Prototype A Complete Guide - 2020 Edition, Gerardus Blokdyk
- Clean Architecture: A Craftsman's Guide to Software Structure and Design, Robert Martin
- Creating and Using Virtual Prototyping Software: Principles and Practices (SEI Series in Software Engineering), Douglass Post, Richard Kendall

Syllabus

A syllabus is provided as a sperate document by the instructor of the course with the following:

- Software Engineering Program Information
- Course Description
- Instructor Contact Information
- Textbook Requirements
- Grading Criteria

- Learning Objectives
- University Policies and Procedures
- Important dates and Deadlines
- Course Policies and Procedures

Communication and Status Updates

Regular, clear, and structured communication with the client, advisor if applicable, and the instructor of the course is critically important to the success of the Capstone project. Students may select an advisor from within the program, the college, the university, or an external advisor from academia, or the industry. Students are expected to maintain regular communication and solicit for input and feedback from all shareholders. For software-intensive projects, regular Git pushes to a public repository (e.g., GitHub) are considered an essential part of demonstrating 'work accomplished' and ensuring transparency of progress. The integrity of the commit history, with no deleted commits, is expected.

To facilitate effective communication, a weekly status report is required. The report will include work accomplished for that period, work planned for next week, and open problems, issues, or obstacles that the student needs help with from the client, advisor, or instructor. The report also includes an overall project status color: Green, all is well on-track, yellow: falling behind at risk, red: derailed off track. Please see Appendix 6 - Capstone Weekly Status Report Template.

Administrative and Academic Policies

The Capstone course is governed by all applicable university policies including the policy below. In addition, several course policies apply. These policies are also be included in the course syllabus. A few critical policies are highlighted below and is of utmost important to the success of the Capstone experience.

This course follows all policies and procedures of UMBC and the University System of Maryland including but not limited to the following policies and procedures:

Academic integrity
Technology Access, Requirements, Resources, Support
Accessibility and Disability Accommodations, Guidance and Resources
Religious Observances & Accommodations
Pregnant and Parenting Students
Sexual Assault, Sexual Harassment, Gender Based Violence and Discrimination
Hate, Bias, Discrimination and Harassment
COVID-19 Response: UMBC Policies and Resources for Students during COVID-19

Academic Integrity

"Academic integrity is an important value at UMBC. By enrolling in this course, each student assumes the responsibilities of an active participant in UMBC's scholarly community in which everyone's academic work and behavior are held to the highest standards of honesty. Cheating, fabrication, plagiarism, and helping others to commit these acts are all forms of academic dishonesty, and they are wrong. Academic misconduct could result in disciplinary action that may include, but is not limited to, suspension or dismissal.
The purposes of higher education are the learning students and faculty undertake, the knowledge and thinking skills developed, and the enhancement of personal qualities that enable students to be strong contributing members of society. In a competitive world, it is essential that all members of the UMBC community uphold a standard that places integrity of each student's honestly earned achievements above higher grades or easier work dishonestly sought." (1)

Academic misconduct includes but not limited to the following acts:

- **Plagiarism**: Knowingly, or unknowingly by negligence, carelessness, or misuse of content is defined by the American Psychological Association (APS) as, "the act of presenting the words, ideas, or images of another as your own; it denies authors or creators of content the credit they are due. Whether deliberate or unintentional, plagiarism violates ethical standards in scholarship." (2)
- **Cheating**: "Using or attempting to use unauthorized material, information, or study aids in any academic exercise." (3)
- **Fabrication**: "Falsification or invention of any information or citation in an academic exercise." (3)
- **Facilitating Academic Dishonesty**: "Helping or attempting to help another commit an act of academic dishonesty." (3)
- **Dishonesty**: "Lack of truthfulness or sincerity when interacting with the faculty member regarding an academic exercise." (3)
- **Use of ChatGPT and AI Tools (4)**
    - If you use ChatGPT (or similar chatbots or AI-based text, image, or video generation tools), you must describe exactly how you used it, including providing the prompt, original generation, and your edits.
    - This applies to prose (text), images (still and video), and code (computer programming code).
    - Not disclosing the use is an academic integrity violation.
    - If you do disclose usage, your answer may receive anywhere from 0 points to full credit, depending on the extent of substantive edits, achievement of learning outcomes, and the overall circumvention of those outcomes.

Academic integrity violations are serious matters. They will be dealt with swiftly:

- First offense: 50% deduction, strong warning, and incident reported to the department.
- Second offense: Zero grade, final warning, and student name placed on academic misconduct list.
- Third offense: Academic misconduct response including but not limited to force withdrawal from the course, F grade in the course, and dismissal proceedings from the program and the college per university academic integrity policy.

(1) Source: https://academicconduct.umbc.edu
(2) Source: https://apastyle.apa.org/style-grammar-guidelines/citations/plagiarism
(3) Source: Courtesy of A. Kleinsmith, HCC629
(4) Source: Adapted from F. Ferraro, CMSC678

Professional Conduct

- Students are expected to conduct themselves in a professional and courteous manner.
- Students must show respect to their classmates, the learning environment, the class TAs, and their instructors.
- Disturbing behavior, loud noise, distractions with mobile devices, and other nonprofessional behavior is not accepted in class.
- Students should raise their hand and wait to be acknowledged to ask questions. When prompted, they should ask their question, and allow for an answer. They should not dominate the class discussion nor interrupt other students.

Class Attendance

- Class attendance for in-person, hybrid, and online classes is required.
- Hybrid classes will have online and in-person class meetings. Announcements on modality of class are sent via the LMS.
- For an in-person class, students are expected to arrive at least 5 minutes before the class start time. For online class, students are expected to join at least 5 minutes before the start time.
- Students are expected to be engaged in class and keep distractions to a minimal.
- Students are expected to keep their devices off and in silent mode.
- Students should join the online session from a quiet space and make sure they are able to engage with the class via audio and video.
- Any disturbing behavior will result in the student being warned and if it continues, the student will be asked to leave the classroom or disconnect from the online class session.
- Students are expected to turn on their camera when the class starts, when asking questions, when interacting with others, during class discussions, before the class ends, and for much of the class time unless otherwise instructed.

- Online classes may require in-person sessions from time to time. One week notice will be given before scheduling an in-person class session. When an in-person class session is scheduled for an online class, attendance is required.

Communication with Your Instructors
Your instructor is here to enable you to be your best. This relationship is built on trust, respect, and the desire to learn and improve. In order to have an effective communication environment, students are expected to use their UMBC email address when corresponding with their instructors. Email messages from personal email addresses can't be replied to and it may be a violation of FERPA. Students may also send messages through the LMS. Please be aware that when sending an email message be specific and provide all the details necessary in your first message in order to answer your question or reply to your request. During university workdays, expect a response within 48 hours to 72 hours depending on your question, the nature of the request, the time of the semester, and the class load. That means if a student sends an email message on Monday at 10:30 pm, the time window starts the next working day Tuesday at 9:00 am and to expect a response by Thursday. During weekends and holidays, your instructor will reply when the university re-opens. Sending multiple messages for the same request is not helpful and creates unnecessary work.

Grading and Final Letter Grade
- Grading is completed within one to two work weeks from the assignment's submission date depending on the class size and the assignment complexity.
- Grading is performed within the LMS.
- Questions on grades given should be directed to the class TA within 3 workdays of the grades being posted. Arithmetic and human errors will be corrected however any challenges to the grading scheme will not be entertained. If a student asks for their grade to be rechecked, the instructor may re-examine all grades that have been given up to that point and make adjustments to the one in question and other past ones based on their findings.
- Final letter grade corresponds to the grading scale in the syllabus.
- The final letter grade awarded is based on the student's achievement in the class, not their effort.

Late work: Homework, project deliverables, and other assignment submissions are only accepted if submitted on time through the course LMS. No submission is accepted via email or other forms. Late work submitted within 24 hours from the due date will receive a 50% penalty. Submissions 24 hours late are not accepted.

Technology use: Students may use their own equipment, university labs and other resources for class work, and third-party computing resources as long as they follow the accepted terms of use, privacy, confidentiality, and other terms of service.

Academic discourse: Good academic discourse is critical to an effective learning environment. Students are expected to communicate with their peers, in class discussions, and outside the class and carry themselves with the highest level of integrity, professionalism, and courtesy. Academic discourse encompasses the entire learning journey including listening, writing, presenting, creating, agreeing, disagreeing, debating, criticizing, and contributing new ideas and knowledge.

Netiquette: Students are expected to conduct themselves on the network and within computing environments as they are expected to conduct themselves in real-life: ethically, professionally, and with great care and respect to others.

Grading: Submitted work are expected to be of high quality, submitted on time, and follow the guidelines and instructions of the assignment to receive a grade. In accordance with university grading policies, final grade letters are awarded based on achievement in the course not effort.

Capstone Phases and Checkpoints

Phase 0: Capstone Proposal

The Capstone proposal is developed and submitted at the beginning of the semester. It serves three purposes:

- It provides a detailed description of the problem, issue, gap, or question that will be addressed by the product, the analysis performed, the data needed, and the schedule for completing major task and milestones. The proposal needs to enable all shareholders, stakeholders, and parties to have a clear understanding of the following:
    - What solution, features, and functions will be provided.
    - Why this project is important and relevant to the client needs.
    - How the project objectives and goals are met and therefore client goals and objectives are met.
    - When interim and final project deliverables will be available.
- The proposal enables all shareholders to have a clear understanding of the project and expected timelines before any significant work has been started and allows for the clarification and correction of any misunderstanding and misalignment before the project is officially launched.
- The proposal serves as a working plan and an agreement among all parties towards the final deliverable, the product, its scope, size, and limitations.

The Elements of a Successful Proposal

A successful proposal starts with understanding the problem. As we often say, software engineering is understanding. This then means that we need to understand the problem, issue, gap, or question sufficiently enough to identify requirements, data, attributes, and subjects that need to be surveyed in order to analyze the problem and a arrive at a solution with usability factors, functions, and features that meet the client needs. Seven critical questions all successful proposals answer:

1. What is the problem, issue, gap, or question that we need to solve for this client? A successful proposal articulates a clear understanding of this and identifies a critical business issue or a critical gap that needs to be filled not simply a side issue, nice to have, or some theoretical interest.
2. What data, information, knowledge, history, and human subjects need to be discovered, evaluated, and analyzed in order to solve the stated problem, remove the issue, fill in the gaps, or answer the question? A successful proposal identifies the most critical past and

current data and subjects that need to be discovered and interviewed to aid in coming with a suitable solution.

3. What engineering requirements methods will be used to collect and analyze the relevant information? A successful proposal specifies the methods used including literature review, multi-level analysis including best practices, market, trends, and competitive analysis, direct data collection through interviews, surveys, and focus groups, indirect data collection through on-site individual and group observation, document analysis, and workflow documentation. The analysis conducted may include a combination of qualitative and quantitative analysis, thematic content analysis, various statistical methods including descriptive and inferential statistics among others.
4. What are the expected outcomes and objectives of this project? In other words, what are the expected outcomes and objectives for this software product? What is the expected impact of this product on users, teams, groups, and organizations using it? What is the expected impact on the market? The community? A successful proposal transforms the problem statement into a set of measurable objectives and expected outcomes.
5. What is the schedule and deliverables at each phase and checkpoint? A successful proposal clearly states the deliverables at each phase and checkpoint with a detailed timeline with dates for milestones, functions, features, and expected usability and quality.
6. What is the level of engagement and collaboration required from the client and other stakeholders? A successful proposal needs to clearly state how much time is required and what involvement is needed from the client and other stakeholders in order for the project to make meaningful progress. The engagement level needs to be clearly stated for each phase in the project.
7. What are the potential opportunities for learning, unlearning, and re-learning from this project? A successful proposal not only identifies the problem and the solution but also looks beyond the delivery of the initial product and identifies potential lessons to be learned during the product development and lessons to be learned beyond this point. What is the future of the product? How will it evolve? How will it be impacted by trends on the horizon?

The Capstone proposal template shows the required sections and information that must be included in the proposal. Students are expected to use the template and submit the proposal to their client, instructor, and advisor for review and approval before starting their project. Two formal consent forms must be completed, accepted, and signed by the student and the sponsor. The student is responsible for securing these forms and ensuring alignment with the sponsor. Please see Appendix 2a and Appendix 2b. In addition, Appendix 3, shows a formal Sample Agreement Between Client and Student.

Quality Criteria

A quality Capstone proposal is a clear expression of the business problem, the proposed requirements analysis, and the desired outcome. In addition, the following quality criteria are offered to guide the student:

- Clarity: The proposal must be clearly written, logically structured, and free of errors.
- Software Engineering Methodology: The proposal must demonstrate student’s knowledge and skills gained from their course work in the program. It must show that they have correctly and accurately applied the relevant principles, methods, techniques, and tools in writing the proposal.
- Structure: The proposal should provide a clear delineation and separation between the problem being solved, the methodology and principles being applied, and the product solution to be designed and built.
- Relevancy: The problems, issue, gap, or question being addressed must be a relevant to modern living and working conditions of individuals, teams, groups, or organizations with a feasible solution through the application of software engineering principles.
- Learning: One of the overarching objectives of a Capstone course is learning including unlearning and relearning. The proposal should demonstrate that the project has the potential for learning and development of new methods, techniques, and tools in the software engineering field.
- Distinct and New Project: Students may choose to conduct a Capstone project with their current employer, however the project may not and cannot be part of their normal work duties. For example, if the student is already working on a software application or a new product prototype for their current employer, they cannot turn this in for their Capstone project. Students are strongly encouraged to seek out projects outside their current organizations as learning to understand a new client and their needs is an important experience, expose the student to a different industry, and increase their professional experience base and their network reach.

Template

A template is provided in Appendix 4 – Capstone Proposal with details on the four sections required and what information needs to be included. Your submission should also include the sponsor contracts detailed above.

Phase 1: Capstone Alpha Checkpoint

Alpha functionality is defined as 50% to 70% of product functionality with most critical functions and features implemented, with known defects and gaps, and adequate quality for non-production use.

Submission must include the following:

1. In progress Capstone Report describing work completed as of this checkpoint. This is only applicable for alpha and beta checkpoints
2. Feedback and review from sponsor on this checkpoint
3. List of functions and features implemented in this checkpoint
4. Source code extracted from the code repository and exported into a zip archive
5. Instructions for how to compile, build, and deploy your application
6. Code statistics including LOC, CLOC, Cyclomatic Complexity, Cohesion, Comment-to-Code Ratio, Coupling, Number of Source Modules, Number of Classes, Number of Methods, Number of Libraries, External Components Used, External Libraries Used, Dependency Tree, and Programming Languages Used
7. Link to code repository
    a. Students are required to use GitHub for their project. Regular commits and pushes are expected to demonstrate continuous progress and active development. The commit history must be preserved in its entirety; students are strictly prohibited from deleting, rewriting, or altering existing commits.
8. Public link to the prototype or a virtual machine image that can be deployed and tested independently
9. Video showing the prototype being compiled, built, and deployed (no longer than 5 minutes at normal playback speed in MP4 format)
10. Video explaining the design, architecture, and the main modules of your code (no longer than 5 minutes at normal playback speed in MP4 format)
11. Video with a demonstration of your prototype functionality (no longer than 5 minutes at normal playback speed in MP4 format)
12. Known problems, gaps, defects, and your plan to address them in the next checkpoint
13. Differences between this checkpoint and the previous one including changes in items 3, 6 and 12. This item is only applicable for beta and final checkpoints

Quality Criteria

Percentage of product functionality implemented.
Percentage of critical functions and features implemented.
Percentage of features and functions with known defects and gaps.
Quality level for non-production use: how long the product can be used without having a crash or a restart, how many users can access the system at the same time before a severe degradation or a system failure?

Phase 2: Capstone Beta Checkpoint

Beta functionality is defined as 70% to 90% of product functionality with all required functions and features implemented, with some known defects and gap, and sufficient quality for limited production use.

Submission should follow the same 13 items as in the Capstone Alpha Checkpoint.

Quality Criteria

Percentage of product functionality implemented.
Percentage of all functions and features implemented.
Percentage of features and functions with known defects and gaps.
Quality level for limited production use: how long the product can be used without encountering a defect, how many users can access the system at the same time before a noticeable downgrade in performance?

Capstone Final Deliverables

The final deliverables of the Capstone project are 4 items:

1. The actual product in the form of a Fully Functional Prototype (FFP), or a Minimally Viable Product (MVP)
2. The 13 items in the Capstone Alpha Checkpoint
3. Capstone report
4. Capstone presentation
5. Client briefing and executive summary

Please find details about each of the deliverables below.

1. Capstone Fully Functional Prototype (FFP) or Minimally Viable Product (MVP)
Your capstone project must have a final deliverable that is either a fully functional prototype or an MVP.

Fully Functional Prototype: Is a proof of concept with sufficient functionality that satisfies the project requirements and the client or sponsor needs. The prototype must use high fidelity UI, implements all of the core features, and the majority of the product functions and features.

MVP (Minimally Viable Product): Is a product that can be taken to the market. It is at least many steps above and beyond a high fidelity fully functional prototype. It should contain all the functions and features required by the target market, low defects count, and should be of sufficient quality for sustained production use. An MVP is typically targeted towards a specific market niche or business domain or users' segment and in some cases may be expanded in scope to appeal to multiple user segments and/or markets.

Quality Criteria for FFP Product

Percentage of critical functions and features implemented: 80% of core functions and features

Percentage of all product functionality implemented: 60% or more

Percentage of features and functions with known defects and gaps: 10%

Quality level for use: how long the product can be used without

encountering a critical defect: 12 hours; increase of memory usage over a period of 12 hours: less than 20% of initial memory usage; what is the average response time for core functions: less than 800ms

Ease of use: Medium to high

Quality Criteria for MVP Product
Percentage of critical functions and features implemented: 90% of core functions and features

Percentage of all product functionality implemented: 70% or more

Percentage of features and functions with known defects and gaps: 10%

Quality level for sustained production use: how long the product can be used without

encountering a critical defect: 24 hours; increase of memory usage over a period of 24 hours: less than 20% of initial memory usage; what is the average response time for core functions: less than 500ms

Ease of use: High

2. Capstone Report
The Capstone report is a detailed documentation of the project from start to finish.

Quality Criteria

- Adheres to the report outline and contains all the sections as in Appendix 5.
- Adheres to the project proposal as approved. If the project deviated from the approved proposal, reasons for this change must be stated and approved by the advisor, the sponsor, and the client in advance of making course changes.
- Each objective in the proposal is addressed even if an objective had to be removed due to an agreed upon conditions by all parties involved.
- Includes a product and literature review of related and similar products.
- Provides detailed descriptions of software engineering methodologies used, including limitations of the project.
- Includes well-supported recommendations and conclusions.
- Demonstrates clarity of writing and thought, well-written, logical flow, well-organized, and free of errors.
- Adheres to APA style format for citations and text format using Times New Roman, 12-point font, and single-spaced.
- Demonstrates a level-of effort and quality appropriate for a graduate capstone project. An e-commerce application or e-commerce website are not acceptable as a capstone

project. Other examples of non-acceptable capstone projects include but not limited to an app without a sponsor, a prototype that does not show critical functionality, an MVP product that is poor in quality and has many defects, an app without clear goal, an app with a poor design and architecture, an app with a poor user interface, and a simple tool that has been done many times over are not acceptable examples. See Appendix 1 for additional acceptable and non-acceptable example projects.

Product and literature review

A required section of the Capstone Report is an APA style product and literature review. Since this section is not based on findings or results of the project, this review is a required deliverable early in the semester and which will eventually be included in the final report. The function of this section is to understand the relevant works for the capstone project that is being undertaken. It should contain a survey and a comprehensive review of existing and related software products that are similar to the current project. This section may contain citations from industry, software engineering research, and related fields.

Report Outline

A detailed report outline is shown in Appendix 5 – Capstone Report Outline.

Lessons Learned and Future Works

Lessons Learned:

Students are required to submit lessons learned paper in a separate document from their Capstone Report. This paper should be written as a reflection of what the student learned through this project and how it has changed their thinking about real-world software engineering.

This paper should be at most three pages long. It should discuss the student's experience with all parties involved and the journey they took from start to finish highlighting main events and lessons learned along the way. The following ten questions should be considered in providing your feedback:

1) What are you most proud of in your Capstone project?
2) What were the main challenges you faced and how did you address them?
3) If you were given the chance to do the project over again, what would you do differently? And why?
4) What is something your instructor, advisor, sponsor, or client have done that helped you significantly in reaching your goals with this project? And what could have they done differently to better support you?
5) What courses in the program were critical to the success of your project?
6) What courses do you wish you had taken that would have made your experience with the capstone more successful?

7) What recommendations do you have for improving the software engineering program?
8) What recommendations do you have for improving the capstone course?
9) What are your plans for your capstone project beyond the end of the semester?
10) What advice would you give to a student who is enrolled in the capstone course in the upcoming semester?

Future works:
This section should explain how the product can be expanded, taken to different markets or users, and possible plans for future works.

3. Capstone Presentation
An oral presentation at the end of the Capstone semester is required. This formal presentation will be given to an audience that may include faculty, advisor, client, sponsor, community members, students, family and friends, and the public.

Quality Criteria
A quality Capstone presentation is a clear and concise expression of the business problem, the actual requirements analysis, software engineering processes and methodologies and a compelling story of findings and recommendations supported by the actual product built. In addition, the following quality criteria are offered to guide the student and serve as the basis for evaluating the product.

The Capstone presentation:

- Provides an accurate and complete explanation of the project. The content is relevant, with a complete description of processes, results, and insights.
- Is clear, logical, and organized.
- Slides are conveniently arranged and follow a logical sequence.
- Speaker uses appropriate eye contact and effective body language. The pace of the presentation is compatible with the audience.
- Slides effectively convey a message, using appropriate spacing, amount of detail, and effective graphics.
- Slides are not crowded with lengthy sentences. Adequate time is allotted per slide.
- Adheres to a recommended time limit of 20 minutes and 20 minutes for discussion and Q&A.

4. Client Briefing
The student is expected to present their findings to the sponsor organization or client as a presentation and in writing as an executive summary providing project overview, results, and future plans.

UMBC

Appendices

Appendix 1 – Acceptable and Non-Acceptable Capstone Project Examples

Non-Acceptable	**Acceptable**
E-commerce website or e-commerce application	A SaaS product that has sponsor specific requirements and provides for new and novel features
Building a website. Example: A website for promoting a non-profit that allows visitors to register and see events and other content that are loaded from a database.	A software application or software intensive product based on a problem that the Greater Baltimore Medical Center is encountering and is looking for a prototype solution to try out.
Building a software application for a problem that has many known and effective solutions. Example: A note taking app.	A software solution for a gap that the UMBC Enrollment Office is encountering that is currently not addressed by the Student Information System (SIS) and the Enterprise Resource System (ERP).
Building a software application without a real-world problem sponsor. Example: An app for tracking books checked out of a library.	A software component for a startup in an emerging field who needs assistance in building an advanced software component.
Building an app based on a project the student already doing at their current job.	A software solution for improving the integration of multiple systems and the visualization of USM tailored actionable insights within the University System of Maryland Office (USM).
Building a software application based on a project the student is already doing in other classes or has done in previous classes.	A software application or software intensive product solution for helping people detach from their hand-held devices that fills in the gaps of the current apps the likes of Freedom, Be Focused Pro, and Self-control in sponsorship and collaboration with research from the UMBC Psychology department.
Building a new software application that requires six months to complete to an alpha quality level.	A software application solution tailored for the needs of the Baltimore City Public Schools to assist in attracting more students to STEM fields.
Building a new software application that requires the purchase of significant hardware and software licenses.	A software application solution for assisting the UMBC library link and unify the multiple systems within the library that is currently not addressed by the solutions provided by the university IT department and the University System of Maryland (USM).

Appendix 2 – Capstone Consent Agreements

To ensure understanding and facilitate the successful execution and completion of the Capstone project, all parties are asked to sign the Capstone consent agreement. The relevant parties are:

- The student
- The sponsor. This person is the Point of Contact (POC) from a business organization, a non-profit entity, or a community group who is authorized to give the student access to people, data, and other resources needed in order for the student to develop the proposal and carry on with the design, architecture, and development of the software product
- The Capstone advisor
- The Capstone instructor

Two consent agreements are needed:

1. Consent Agreement, Discovery - The first consent agreement must be reviewed and signed by all parties prior to the development of the Capstone Proposal. The purpose of this agreement is to obtain authorization and approval from the sponsor to do the work at the organization. This enables the proposal development process to proceed with clear understanding of the role of each party during proposal development and subsequent phases.
2. Consent Agreement, Proposal - The second consent agreement includes a detailed proposal agreed to by all parties.

Appendix 2a – Capstone Consent Agreement - Discovery

Student name and contact information:
Capstone advisor name and contact information:
Sponsor organization name, sponsor address, sponsor point of contact name and contact information
Proposed Project Title:
Stakeholder Expectations: A successful capstone project requires support and engagement from the student, instructor, advisor, and the sponsor organization. The parties signing below agree to the following roles and responsibilities.

Student agrees to:

- a formal Capstone proposal that reflects the student understanding of the problem, its relevance to Software Engineering, the objectives, and a proposed method and timeline.
- a recognition that real world projects are messy and unpredictable, and, therefore, a willingness to re-negotiate and revise the project as necessary.
- a finished Final Report, formal briefing, and a software product with the agreed to functionality and quality that meets the sponsor organization requirements and expectations.

Sponsor agrees to:

- a concrete, feasible project that facilitates learning for the student.
- granting the student access to all of the materials, data, and staff required for the project including a weekly real-time meeting with the student
- a willingness to review and provide input to the student, advisor, and faculty instructor as necessary throughout the project from initial proposal to final deliverable.

SENG Capstone instructor agrees to:

- a learning environment in which students can build on earlier experiences to create a significant software intensive product.
- intellectual and technical expertise and experience.
- communication with the sponsor to ensure that the project is moving forward and act as liaison, between UMBC, the sponsor and the student.

Student Name (print) Signature Date

____

Sponsor Name (print) Signature Date

____

Advisor Name (print) Signature Date

____

Capstone Instructor Name (print) Signature Date

____

Appendix 2b – Capstone Consent Agreement - Proposal

Student name and contact information:
Capstone advisor name and contact information:
Sponsor organization name, sponsor address, sponsor point of contact name and contact information
Proposed Project Title:
Stakeholder Expectations: A successful capstone project requires support and engagement from the student, instructor, advisor, and the sponsor organization. The parties signing below agree to the following roles and responsibilities.

Student agrees to:

- a formal Capstone proposal that reflects the student understanding of the problem, its relevance to Software Engineering, the objectives, and a proposed method and timeline.
- a recognition that real world projects are messy and unpredictable, and, therefore, a willingness to re-negotiate and revise the project as necessary.
- a finished Final Report, formal briefing, and a software product with the agreed to functionality and quality that meets the sponsor organization requirements and expectations.
- A detailed proposal is provided along with a time schedule

Sponsor agrees to:

- a concrete, feasible project that facilitates learning for the student.
- granting the student access to all of the materials, data, and staff required for the project including a weekly real-time meeting with the student
- a willingness to review and provide input to the student, advisor, and faculty instructor as necessary throughout the project from initial proposal to final deliverable.
- Have reviewed and agreed to the proposal and the time schedule provided by the student.

SENG Capstone instructor agrees to:

- a learning environment in which students can build on earlier experiences to create a significant software intensive product.
- intellectual and technical expertise and experience.
- communication with the sponsor to ensure that the project is moving forward and act as liaison, between UMBC, the sponsor and the student.
- Have approved the proposal and the time schedule provided by the student.

Student Name (print) Signature Date

____

Sponsor Name (print) Signature Date

____

Advisor Name (print) Signature Date

____

Capstone Instructor Name (print) Signature Date

____

Appendix 3 – Sample Agreement Between Client and Student

This is an agreement between [name of sponsor here], the "Client", and the Software Engineering student [name of student here], the "Student", from the [type semester and year here] In the Software Engineering Capstone SENG 701 class, the "Class", at the University of Maryland Baltimore County (UMBC), the "University", with Dr. Mohammad Samarah, the "Professor", concerning designing and building a software application for the as part of the Class project to be used by the Client in service of their own clients and patrons, the "Software". The agreement is valid from [type start date here], the "Start Date", through [type end date here], the "End Date". The following are the terms and conditions of this agreement:

1. This agreement is exclusively between the Student of the Class and the Client and neither the Professor nor the University are party to it. The Client agrees to not hold the Professor and the University liable to any damages or loss of business, wages, or any other damages real or perceived.
2. The Software is being provided free of charge to the Client by the Student.
3. This Software will be given to you in its entirety, including the Software source code. You may modify, enhance, or reuse the Software for your own use. You may not redistribute the Software in any form or provide it to other entities or organizations without written permission from the Student.
4. The Student holds exclusive rights to the Software, its design, and its source code. The Client may not publish the Software source code in any form.
5. The Student will make every effort to provide a highly usable Software. You agree and understand that certain functions and features may not be available or missing and agree to not hold the Student liable for the completion of these functions and features.
6. Security is a major consideration in designing this Software, and great efforts by the Student will be taken to minimize the risk of data leakage, including security-driven design, code reviews, and the use of best practice software engineering methods. You agree to not hold the Student liable for any damages caused by Software exploitations, security attacks or any other Software omissions, errors, or failures.
7. This Software will be provided as is in beta or release candidate quality, not as a final production quality release. It may contain bugs, defects, and missing features and functions. For example, dropped events and failure to send emails and notifications. Client agrees to not hold Student liable for any losses or damages caused by Software omissions, errors, and failures.
8. The Software does not include any warranties of any sort.
9. No updates, support, or any other services are provided after the End Date. Further work on the Software requires a separate agreement between the Client and the Student.
10. If there are disagreements or contradictions between this agreement and the UMBC intellectual property rights policy, then the later policy is in effect.

Appendix 4 – Capstone Proposal Template

The capstone proposal has four sections each with its own information. Section I is project data, section II is project information, and section III is project background. Each of these sections appears only once in the proposal. Section V is the proposed solution which provides statement of functional and non-Functional requitement sets and may repeat multiple times. The proposal must have at a minimum of three sets in section V. Each set describes one or more significant function, feature, or system or application capability.

Section 1: Project Data

Student Information

- Student name:
- Contact phone number:
- UMBC email address:
- Semester and year of capstone experience:
- Expected graduation date:

Capstone Course Information

- Capstone faculty (faculty teaching and supervising the capstone course):
- Capstone advisor:

Sponsor Client Information

- Client contact name:
- Client contact title:
- Client contact phone number:
- Client contact email address:
- Client organization name:
- Client organization other stakeholders with interest in this project and their titles:

Section 2: Project Information

- Project title
- Problem statement: clear and concise description of the client problem, issue, gap, or question to be solved and addressed by this project.
- Short description of project history and evolution including how this project opportunity became available with this client, what was the original problem or request, and how the project scope evolved as the client needs were discovered and understood.

Section 3: Project Background

- Description of client and their organization including organizational chart.
- Description of the client, stakeholders, shareholders, and their expectations for this project.
- Description of required resources and where they will be obtained.
- Anticipated challenges, risks, and mitigation strategies.

Section 4: Proposed Solution

The requirements should be stated in outcome format as to the benefit or problem it solves. For example, ability to securely login to the system from any device. For each requirements set, describe the following:

- Requirements engineering methods used to understand and analyze the requirement.
- How the functions, features, and system abilities are analyzed, prototyped, and built.
- What data is needed, how the data is collected, secured, visualized, and transformed into actionable insights that benefit the end user.
- Expected benefits of function, feature, or performance level of non-functional requirements.
- The requirement must be stated as a user story in one of the following formats:
    - As a [type of user], I want to be able to do this [action], so that I achieve this [benefit or value]
    - As a [type of user], I want this [capability], so that I achieve [this benefit]
    - As a [type of user], I want this [function or feature], so that I achieve [this goal or objective]
- Each user story must be included in the low-fidelity or high-fidelity UI prototype showing the user interface and the interactions between this user story and other user stories. The included prototype should be a cohesive start-to-finish experience of your actual product with minimal focus on account creation and logging in as these are standard features.
- The proposal must have at a minimum 12 user stories of varying complexity.

Section 5: Git Repository Link

A Git repository must be created, and a link to it provided, before the Capstone Proposal is accepted. This single repository will serve as the official repository for all deliverables submitted for proposal, alpha, beta, and final checkpoints.

Statement of Functional and Non-Functional Requitements and User Stories:

Req. ID	Requirement stated as a user story	Expected Completion Date	Complexity (High, medium, Low)	Risk (High, medium, low)
R1				
R2				
R3				
R4				

Statement of Functional and Non-Functional Requitements and User Stories:

Req. ID	Requirement stated as a user story	Expected Completion Date	Complexity (High, medium, Low)	Risk (High, medium, low)
R5				
R6				
R7				
R8				

UMBC

Statement of Functional and Non-Functional Requitements and User Stories:

Req. ID	Requirement stated as a user story	Expected Completion Date	Complexity (High, medium, Low)	Risk (High, medium, low)
R9				
R10				
R11				
R12				

Appendix 5 – Capstone Report Outline

Below is a list of sections for the Capstone project report. The report should be written in past tense and in APA format. There are three major parts to the report: part I, the background section is section 1 through 5, part II, the core product design and implementation is section 6 through 9 and part III, action items and reflection is section 10 through 12.

The following section headings as well as subsections should be used for organization and clarity.

Part I: The background
1. Title page
2. Table of contents
3. Abstract
4. Introduction
    a. Purpose of the project
    b. Problem statement
    c. Background of sponsor organization
    d. Background of Problem (including description from the sponsor and their needs)
5. Product review of existing similar ideas that can be a product, an application, or a trend from the industry or academia

Part II: The core product design and implementation
6. Detailed requirements - Functional and Non-functional (this section should include capabilities, features, and functions of the product as well as user stories, lines of code, and other data.)
7. Design, architecture, and methodology - To include subsections for software engineering methodology and methods used, high level and low-level design, system architecture, hardware and software dependencies and detailed components design
8. Results and discussion
9. Conclusion

Part III: Action items and reflection
10. Recommendations to the sponsor to include workflow changes, best way of deploying the product for their environment and a detailed user manual.
11. Limitations of the project or approach
12. Future works
13. References

Appendix 6 – Capstone Weekly Status Report Template and Example

Date: August 6, 2023
Student Name: <student name>Kathy Longwood
Capstone Project: <capstone project name>STEM Engagement for K6-K12
Capstone Sponsor: <sponsor organization and contact person>Baltimore Public School District, Zakiya Lee
Capstone Advisor: <advisor organization and contact person>Sanjoy Khan, UMBC Education department

Overall Status

- Project Status Color: <Green, Yellow, Red>Green
- Project Status Color from Previous Week: <Green, Yellow, Red>NA (this is the first status report)

Status Details

Work accomplished this week (artifacts and documents are stored in GitHub repository):

- Created two use cases based on initial interview with client.
- Created a drawing of first screen.
- Compiled a list of most needed functions.

Work planned for next week:

- Create remaining use cases.
- Have a complete prototype.
- Have an agreed to list of functions.

Problems, obstacles, needs, or questions that I need help with from client, advisor, or instructor or need to bring to their awareness with no actions on their part:

- Need to continue to get clarity on engagement modality while working with staff in downtown office. Having difficulty getting time scheduled and working through this issue (no actions for client at this time). This is just for your awareness.
- Need to meet with a K10 teacher and need help with client to get this scheduled for next week.

Appendix 7 – Capstone Sponsor Criteria

The sponsor must meet the following criteria:

- A company, a non-profit, or a registered organization in the tri-state area (DC-VA-MD) from the commercial, educational, non-profit, or government sectors. The sponsor also may be an office, department, or a student organization within UMBC
- Has a problem, issue, gap, or question that can be clearly defined and that a software application or a software-intensive device is a very good and feasible solution
- Has been established and in operation for at least 12 months with three or more employees
- The sponsor line of work adheres to the limitations and exclusions including but not limited to no weapon making machines or systems, gambling, addictive applications, and applications with sexual or racial content
- Have physical offices and presence in the tri-state area
- Not related to the student through employment, familial, friendship, or other employment and personal relationships
- Willing and available to meet with the course instructor or their delegate in person or via a video meeting once before the proposal is accepted and another time before the final checkpoint
- The sponsor selection must be approved by the course instructor and the department before approaching the sponsor

Appendix 8 – Checklists

Proposal Deliverable Checklist

Please see relevant pages and Appendix 4 for further guidance on what to include.

Your proposal must include the major items listed below:

- Student information
- Capstone course information
- Sponsor information
- Project information
- Project background
- Proposed solution including these five items:
    - Requirement methods
    - Methods for prototyping, building, and analyses
    - Data needed and data management plan
    - A minimum of twelve (12) user stories using the table format in Appendix 4 with functional and non-functional requirements
    - Lo-fi or hi-fi prototype that shows a complete and cohesive experience of using all aspects the product
- Git Repository Link
- Conclusion or summary

Alpha Deliverable Checklist

Please see other sections of the Capstone Handbook for further details and guidance. This list is provided for your convenience and to ensure that all items are completed and submitted. Please ensure all items are checked off before submission.

Alpha functionality is defined as 50% to 70% of product functionality with most critical functions and features implemented, with known defects and gaps, and adequate quality for non-production use.

Your alpha submission must include the five major items below:

- Percentage of product functionality implemented.
- Percentage of critical functions and features implemented.
- Percentage of features and functions with known defects and gaps.
- Quality level for non-production use: how long the product can be used without having a crash or a restart, how many users can access the system at the same time before a severe degradation or a system failure?
- The 12 alpha capstone checkpoint items listed below:
    1. In-progress Capstone Report describing work completed as of this checkpoint.

2. Feedback and review from your sponsor on this checkpoint. Email correspondence with your sponsor, if applicable, is sufficient.
3. List of functions and features implemented in this checkpoint. This should be a section in the report.
4. Source code extracted from the code repository and exported into a zip archive. Provide as a separate zip archive named *FirstName*_LastName_final_source_code_repository.zip
5. Instructions for how to compile, build, and deploy your application. This should be an appendix in the report.
6. Code statistics including but not limited LOC, CLOC, cyclomatic complexity, cohesion, comment-to-code ratio, coupling, number of source modules, number of classes, number of methods, number of libraries, external components used, external libraries used, dependency tree, and programming languages used. This should be an appendix in the report.
7. Link to code repository. Ensure regular Git pushes are made to show continuous work, and no commits are deleted or altered in the repository history.
8. Public link to your application or a virtual machine image that can be deployed and tested Independently. This should be in the report under a section titled Final Deliverables Access.
9. Video showing the application being compiled, built, and deployed (no longer than 5 minutes at normal playback speed in MP4 format). This is provided as a separate file named FirstName_LastName_compile_build_deploy.mp4
10. Video explaining the design, architecture, and the main modules of your code (no longer than 5 minutes at normal playback speed in MP4 format). This is provided as a separate file named FirstName_LastName_design_architecture_modules.mp4
11. Video with a demonstration of your application functionality (no longer than 5 minutes at normal playback speed in MP4 format). This is provided as a separate file named FirstName_LastName_product_demo.mp4
12. Known problems, gaps, defects, and your plan to address them in the future works if applicable. This should be included as a section in your report.

Example submission would include 6 separate files: 2 PDFs, 1 ZIP, and 3 MP4s. Please use the naming convention below to name your files:

- Shourya_Rami_alpha_report.pdf
- Shourya_Rami_alpha_presentation.pdf
- Shourya_Rami_alpha_source_code_repository.zip
- Shourya_Rami_compile_build_deploy.mp4
- Shourya_Rami_design_architecture_modules.mp4
- Shourya_Rami_product_demo.mp4

Beta Deliverable Checklist

Please see other sections of the Capstone Handbook for further details and guidance. This list is provided for your convenience and to ensure that all items are completed and submitted. Please ensure all items are checked off before submission.

Beta functionality is defined as 70% to 90% of product functionality with all required functions and features implemented, with some known defects and gap, and sufficient quality for limited production use.

Your beta submission must include the five major items below:

- Percentage of product functionality implemented.
- Percentage of critical functions and features implemented.
- Percentage of features and functions with known defects and gaps.
- Quality level for limited production use: how long the product can be used without having encountered a defect, how many users can access the system at the same time before a noticeable downgrade in performance?
- The 13 beta capstone checkpoint items listed below:
    1. In-progress Capstone Report describing work completed as of this checkpoint.
    2. Feedback and review from your sponsor on this checkpoint. Email correspondence with your sponsor, if applicable, is sufficient.
    3. List of functions and features implemented in this checkpoint. This should be a section in the report.
    4. Source code extracted from the code repository and exported into a zip archive. Provide as a separate zip archive named *FirstName*_LastName_final_source_code_repository.zip
    5. Instructions for how to compile, build, and deploy your application. This should be an appendix in the report.
    6. Code statistics including but not limited LOC, CLOC, cyclomatic complexity, cohesion, comment-to-code ratio, coupling, number of source modules, number of classes, number of methods, number of libraries, external components used, external libraries used, dependency tree, and programming languages used. This should be an appendix in the report.
    7. Link to code repository. Ensure regular Git pushes are made to show continuous work, and no commits are deleted or altered in the repository history.
    8. Public link to your application or a virtual machine image that can be deployed and tested Independently. This should be in the report under a section titled Final Deliverables Access.
    9. Video showing the application being compiled, built, and deployed (no longer than 5 minutes at normal playback speed in MP4 format). This is provided as a separate file named FirstName_LastName_compile_build_deploy.mp4

10. Video explaining the design, architecture, and the main modules of your code (no longer than 5 minutes at normal playback speed in MP4 format). This is provided as a separate file named FirstName_LastName_design_architecture_modules.mp4
11. Video with a demonstration of your application functionality (no longer than 5 minutes at normal playback speed in MP4 format). This is provided as a separate file named FirstName_LastName_product_demo.mp4
12. Known problems, gaps, defects, and your plan to address them in the future works if applicable. This should be included as a section in your report.
13. Differences between this checkpoint and the previous one including changes in items 3, 6, and 12. This should be in an appendix in the report.

Example submission would include 6 separate files: 2 PDFs, 1 ZIP, and 3 MP4s. Please use the naming convention below to name your files:

- Shourya_Rami_beta_report.pdf
- Shourya_Rami_beta_presentation.pdf
- Shourya_Rami_beta_source_code_repository.zip
- Shourya_Rami_compile_build_deploy.mp4
- Shourya_Rami_design_architecture_modules.mp4
- Shourya_Rami_product_demo.mp4

Final Deliverable Checklist

Please see other sections of the Capstone Handbook for further details and guidance. This list is provided for your convenience and to ensure that all items are completed and submitted. Please ensure all items are checked off before submission.

Your final submission must include the five major items below:

- Your final capstone report as a separate PDF document and your lessons learned paper
- as a separate PDF document (as described on page 18 and Appendix 5 of the Capstone Handbook).
- Your final capstone presentation as a PDF file.
- Your client briefing which includes an executive summary at the top as a separate PDF document (as described on page 19 of the Capstone Handbook).
- Your product. This is your fully functional prototype (FFP) or minimally viable product (MVP). This must be accessible via a public URL or provided as a virtual machine (VM) or VM container or a remotely accessible machine with login and access details for the TA and instructors. Your application must remain live on a publicly accessible URL without any changes from submission until the course closes. This should be in the report under a section titled Final Deliverables Access
- The 13 final capstone checkpoint items listed below:
    1. Capstone report: This is your final full capstone report
    2. Feedback and review from your sponsor on this checkpoint. Email correspondence with your sponsor, if applicable, is sufficient

3. List of functions and features implemented in this checkpoint. This should be a section in the report
4. Source code extracted from the code repository and exported into a zip archive. Provide as a separate zip archive named FirstName_LastName_final_source_code_repository.zip
5. Instructions for how to compile, build, and deploy your application. This should be an appendix in the report
6. Code statistics including but not limited LOC, CLOC, cyclomatic complexity, cohesion, comment-to-code ratio, coupling, number of source modules, number of classes, number of methods, number of libraries, external components used, external libraries used, dependency tree, and programming languages used. This should be an appendix in the report.
7. Link to code repository. Ensure regular Git pushes are made to show continuous work, and no commits are deleted or altered in the repository history.
8. Public link to your application or a virtual machine image that can be deployed and tested Independently. This should be in the report under a section titled Final Deliverables Access.
9. Video showing the application being compiled, built, and deployed (no longer than 5 minutes at normal playback speed in MP4 format). This is provided as a separate file named FirstName_LastName_compile_build_deploy.mp4
10. Video explaining the design, architecture, and the main modules of your code (no longer than 5 minutes at normal playback speed in MP4 format). This is provided as a separate file named FirstName_LastName_design_architecture_modules.mp4
11. Video with a demonstration of your application functionality (no longer than 5 minutes at normal playback speed in MP4 format). This is provided as a separate file named FirstName_LastName_product_demo.mp4
12. Known problems, gaps, defects, and your plan to address them in the future works if applicable. This should be included as a section in your report.
13. Differences between this checkpoint and the previous one including changes in items 3, 6 and 12. This should be in the report in an appendix.

Example submission would include 8 separate files: 4 PDFs, 1 ZIP, and 3 MP4s. Please use the naming convention below to name your files:

- Shourya_Rami_final_report.pdf
- Shourya_Rami_lessons_learned.pdf
- Shourya_Rami_client_briefing.pdf
- Shourya_Rami_final_presentation.pdf
- Shourya_Rami_final_source_code_repository.zip
- Shourya_Rami_compile_build_deploy.mp4
- Shourya_Rami_design_architecture_modules.mp4
- Shourya_Rami_product_demo.mp4