CS 481W/482: Senior Capstone Project I & II

Computer Science Capstone

Lucas P. Cordova, Ph.D.

2025-08-28

Table of contents

1 Course Information

1.1 Basic Information

Component	Details
Course Numbers	CS 481W (Fall 2025) / CS 482 (Spring 2026)
Credits	4 credits per semester (8 total)
Duration	Two semesters (30 weeks)
Meeting Format	Weekly individual meetings (30 minutes)
Prerequisites	Senior standing; completion of core CS curriculum

1.2 Professor

Lucas Cordova, Ph.D. **Email:** LPCordova@willamette.edu **Office:** Ford 210 (Salem)

1.3 Office Hours

Office hours are available by 15-minute appointments or on a drop-in basis when available. Multiple modalities are offered (in-person, phone, Google Meet). If the scheduled times do not align with your availability, please contact the instructor.

1.4 Course Description

A two-semester culminating experience where students apply their cumulative knowledge to design, develop, and deploy a full-stack or similar-type application. Students will follow industry-standard software development lifecycle (SDLC) practices, from initial conception through final deployment and documentation. The course emphasizes professional software engineering methodology, project management, technical communication, and collaborative development practices.

1.5 Learning Objectives

Upon completion of this course sequence, students will be able to:

1. Design and implement a complete full-stack or similar-type application from conception to deployment
2. Apply software engineering principles and best practices throughout the development lifecycle
3. Demonstrate proficiency in project management and agile methodologies
4. Create comprehensive technical documentation
5. Present technical work to both technical and non-technical audiences
6. Collaborate effectively using version control and project management tools
7. Conduct meaningful peer reviews and incorporate feedback

1.6 Textbooks and Materials

Material	Details	Required
Software Engineering: A Modern Approach	Marco Tulio Valente. Free, open textbook. Available at engsoftmoderna.info	Required
Handbook of Software Engineering Methods	Lara Letaw. Free OER via Open Textbook Library. Available at open.umn.edu	Required

Software Engineering: A Modern Approach provides comprehensive coverage of software engineering topics including process models, requirements, architecture, design, testing, and maintenance. *Handbook of Software Engineering Methods* complements it with a practical, methods-oriented framing aligned with real-world project work. Both texts are free and open-access.

2 Software Development Methodology

2.1 Modified Agile with Academic Milestones

This course employs a Modified Agile methodology that combines industry-standard Agile practices with academic milestone requirements.

2.1.1 Core Principles

1. **Iterative Development:** Features built incrementally with regular feedback loops
2. **Weekly Sprints:** One-week development cycles aligned with instructor meetings
3. **Continuous Integration:** Code integrated and tested regularly
4. **Adaptive Planning:** Requirements evolve based on feedback and learning
5. **Stakeholder Engagement:** Instructor as product owner (or external customer in certain situations) and peers as stakeholders

2.2 Development Cycle Structure

2.2.1 Weekly Sprint Cycle

The following provides a high-level overview of the weekly sprint cycle. The precise schedule may vary based on project needs and progress.

Day	Activity	Deliverable
Monday	Sprint planning	Sprint backlog
Tuesday–Thursday	Development work	Code commits
Friday	Code review and testing	Pull requests
Weekend	Documentation and reflection	Updated docs
Weekly Meeting	Sprint review with instructor	Demo

2.2.2 Scrum Artifacts (Academic Adaptation)

1. **Product Backlog:** Living document in GitHub Issues
2. **Sprint Backlog:** Weekly task list with estimates
3. **Burndown Charts:** Visual progress tracking
4. **Definition of Done:** Explicit completion criteria

2.3 Required Development Practices

2.3.1 Version Control Requirements

- Daily commits minimum (including research/documentation days)
- Commit format: Conventional commits (`feat:`, `fix:`, `docs:`, etc.)
- Workflow: Feature branch with pull requests
- Protection: Main branch requires review

2.3.2 Testing Strategy

Test Type	Requirement	Coverage Target
Unit Tests	All public methods	60% minimum (Semester 2)
Integration Tests	All API endpoints	100%
End-to-End Tests	Critical user paths	Core features
Performance Tests	Database queries	$<$ 100ms response

2.3.3 CI/CD Pipeline

A continuous integration and continuous deployment (CI/CD) pipeline will be established to automate testing and deployment processes, ensuring rapid and reliable delivery of software updates. The implementation will vary based on the project's technical stack.

Example workflow configuration for GitHub Actions:

```yaml
name: CI/CD Pipeline
on: [push, pull_request]
jobs:
  test:
    - run: npm test
    - run: npm run coverage
  deploy:
    - staging: automatic
    - production: manual approval
```

2.4 Agile Ceremonies

2.4.1 Sprint Planning (Weekly – 10 min)

- Review velocity
- Select backlog items
- Break down tasks
- Commit to goals

2.4.2 Daily Standup (Asynchronous)

Post daily by 10 AM in Discord/Slack:

1. Yesterday's completions

2. Today's plan
3. Current blockers

2.4.3 Sprint Review (Weekly – 15 min)

- Demo features
- Gather feedback
- Discuss incomplete items

2.4.4 Sprint Retrospective (Bi-weekly – 5 min)

- What went well
- Areas for improvement
- Action items

3 Course Structure

3.1 Semester 1: Planning, Design, and Foundation (CS 481W – Fall 2025)

3.1.1 Milestone 1: Project Initiation and Planning (Weeks 1–4)

Week	Topic	Deliverable
1	Course Introduction and Project Brainstorming: course overview, SDLC methodologies review, project ideation workshop	Three project ideas (one page each)
2	Project Selection and Feasibility: technology stack research, feasibility analysis, market research	Project selection with justification
3	Project Proposal Development: scope definition, user stories, risk assessment	Draft project proposal

Week	Topic	Deliverable
4	Proposal Finalization: proposal refinement, timeline planning, methodology selection	Final project proposal (5–7 pages)

3.1.2 Milestone 2: Design and Architecture (Weeks 5–8)

Week	Topic	Deliverable
5	System Architecture Design: high-level architecture, component diagrams, database schema	Architecture diagram
6	UI/UX Design: wireframes, user flows, accessibility planning	Complete wireframe set
7	Detailed Design: API specifications, data flow diagrams, security architecture	Technical design document
8	Mid-Semester Presentation: present to instructor and peers, peer feedback, Q&A	15-minute presentation

3.1.3 Milestone 3: Foundation Development (Weeks 9–14)

Week	Focus Area	Key Deliverable
9	Environment Setup	CI/CD pipeline
10	Database Implementation	Schema with test data
11	Backend Foundation	Auth system and core APIs
12	Frontend Foundation	Basic UI with routing
13	Integration	Working prototype
14	Documentation	Complete code docs

3.1.4 Milestone 4: Semester 1 Wrap-up (Week 15)

- Working prototype demonstration
- Semester retrospective
- Semester 2 planning
- **Deliverable:** Prototype demo and reflection paper

3.2 Semester 2: Development, Testing, and Deployment (CS 482 – Spring 2026)

3.2.1 Milestone 5: Feature Development (Weeks 1–6)

- **Week 1:** Sprint planning and setup
- **Weeks 2–3:** Core Feature Sprint 1
- **Weeks 4–5:** Core Feature Sprint 2
- **Week 6:** Integration testing and fixes

3.2.2 Milestone 6: Testing and QA (Weeks 7–10)

Week	Activity	Focus
7	Test Planning	Test cases, scripts
8	Mid-Semester Presentation	Technical demo
9	User Acceptance Testing	5+ user sessions
10	Refinement Sprint	Priority fixes

3.2.3 Milestone 7: Deployment and Documentation (Weeks 11–13)

- **Week 11:** Production environment setup
- **Week 12:** Deployment and monitoring
- **Week 13:** Final documentation package

3.2.4 Milestone 8: Project Closure (Weeks 14–15)

- **Week 14:** Showcase preparation
- **Week 15:** Public presentation and handoff

4 Peer Review Process

4.1 Overview

Each student serves as a peer reviewer for two projects throughout both semesters, developing critical evaluation skills and exposure to different approaches.

4.2 Review Partnerships

Review partnerships are assigned at the start of Semester 1 and remain consistent throughout both semesters. Each student provides constructive feedback and support to their assigned peers.

4.3 Review Schedule

4.3.1 Semester 1 Reviews

1. Week 3: Proposal Draft
2. Week 6: Architecture
3. Week 8: Design Docs
4. Week 11: Backend Code
5. Week 13: Frontend Code
6. Week 15: Prototype

4.3.2 Semester 2 Reviews

1. Week 2: Sprint Planning
2. Week 5: Features
3. Week 8: Presentation
4. Week 9: Testing
5. Week 11: Deployment
6. Week 12: Documentation
7. Week 14: Final Code
8. Week 15: Showcase

4.4 Review Rubric

Grade	Criteria	Point Range
A	Specific, actionable feedback with code examples	90–100%
B	Identifies major issues with useful suggestions	80–89%
C	Meets requirements with basic feedback	70–79%
D	Minimal effort, vague feedback	60–69%
F	Missing or non-constructive	Below 60%

5 Assessment and Grading

5.1 Grade Scale

Percentage	Grade	Percentage	Grade
92.00	A	72.00 – 77.99	C
90.00 – 91.99	A-	70.00 – 71.99	C-
88.00 – 89.99	B+	68.00 – 69.99	D+
82.00 – 87.99	B	62.00 – 67.99	D
80.00 – 81.99	B-	60.00 – 61.99	D-
78.00 – 79.99	C+	59.99	F

5.2 Semester 1 Grade Distribution (CS 481W)

Component	Weight	Description
Project Proposal	15%	Complete proposal document
Design Documentation	15%	Architecture and design docs
Weekly Progress	20%	Meeting attendance and progress
Mid-Semester Presentation	10%	Design presentation
Working Prototype	25%	Functional MVP
Final Presentation	10%	Semester wrap-up
Peer Reviews	5%	Quality of reviews provided

5.3 Semester 2 Grade Distribution (CS 482)

Component	Weight	Description
Feature Completion	25%	All planned features implemented
Testing and Quality	15%	Test coverage and bug fixes

Component	Weight	Description
Deployment Success	15%	Production deployment
Documentation	15%	User and technical docs
Weekly Progress	15%	Consistent development
Final Presentation	10%	Project showcase
Peer Reviews	5%	Quality of reviews provided

6 Required Tools and Resources

6.1 Technology Stack

- **Version Control:** Git/GitHub (required)
- **Project Management:** Jira, Trello, or GitHub Projects
- **Communication:** Discord or Slack
- **CI/CD:** GitHub Actions, Jenkins, or CircleCI
- **Cloud:** Student credits for AWS/Azure/GCP
- **Documentation:** Markdown, JSDoc, or Sphinx

6.2 Development Environment

```
# Minimum required setup
git --version      # 2.0+
node --version     # 14.0+ (if using Node.js)
docker --version   # 20.0+ (recommended)
```

7 Project Proposal Template

7.1 Executive Summary (1 page)

- Project name and tagline
- Problem statement
- Proposed solution
- Target users

7.2 Technical Specifications (2–3 pages)

- Functional requirements (minimum 10)
- Non-functional requirements
- Technology stack justification
- System architecture overview

7.3 Project Management (1–2 pages)

- Development methodology
- Timeline with milestones
- Risk analysis matrix
- Success metrics

7.4 References and Research (1 page)

- Similar solutions analysis
- Technical resources
- Learning requirements

8 Course Policies

8.1 Academic Integrity

All code must be original or properly attributed. AI tool usage must be documented. Plagiarism results in course failure. For details, consult Willamette's academic integrity policy.

8.2 Late Work

Students are allotted 72 cumulative hours (3 days) of late submissions across all deliverables without penalty. Hours may be distributed across assignments as needed. Once exhausted, late submissions incur a 20% penalty per 24 hours on a continuous scale; deliverables submitted more than 5 days late receive no credit. Extenuating circumstances should be discussed with the instructor promptly.

8.3 Incomplete Grades

Incomplete grades are granted only for prolonged illness or family emergencies that remove a student from the learning environment for an extended period. Incompletes are not granted for falling behind due to motivation, comprehension, or time management difficulties. Students experiencing academic difficulties should consult with the instructor to develop an improvement plan.

8.4 Time Commitment

Per Willamette's Credit Hour Policy, each hour of class time requires 2–3 hours of work outside class. The advanced nature of this course requires consistent effort throughout the semester. Expect 6–9 hours per week of outside engagement including development, documentation, and project work.

8.5 Classroom Conduct

Constructive classroom behavior supports an environment of trust, respect, and collaborative learning. Disruptive behaviors include but are not limited to: interrupting others, distracting from course content, unauthorized recordings, and any form of harassment or abuse. Such behaviors will not be tolerated.

8.6 Accommodations

Students requiring accommodations should provide documentation from Accessible Education Services (Smullin 155, 503-370-6737, accessible-info@willamette.edu) within the first two weeks.

8.7 Office Hours

- **By Appointment:** Project-specific consultations
- **Online:** Discord server for async questions

9 Appendix: Additional Resources

9.1 Recommended Reading

1. *The Pragmatic Programmer* by Hunt and Thomas
2. *Clean Code* by Robert Martin
3. *Design Patterns* by the Gang of Four

9.2 Online Resources

- GitHub Docs
- MDN Web Docs
- Stack Overflow
- DevDocs

9.3 Success Tips

1. Start early and iterate often
2. Commit code daily, even small changes
3. Ask for help when stuck for more than 2 hours
4. Attend all weekly meetings prepared
5. Engage meaningfully in peer reviews
6. Document as you go, not at the end

10 University Policies

10.1 Inclusive Classroom

Students are addressed by their affirmed name and pronouns upon request. Notify the instructor at any point to update records accordingly.

10.2 Accessibility and Accommodations

Willamette University is committed to creating inclusive learning environments. Students experiencing barriers to inclusion or achievement should notify the instructor promptly. Students with disabilities are encouraged to contact Accessible Education Services (Smullin 155, 503-370-6737, accessible-info@willamette.edu) to discuss accommodations.

10.3 Title IX

Willamette University prohibits discrimination and harassment based on sex or gender. As a mandatory reporter, the instructor is required to report any incidents of sexual misconduct disclosed to them to Willamette's Title IX Coordinator. For confidential support, contact:

- Confidential advocate: confidential-advocate@willamette.edu
- WUTalk crisis line: 503-375-5353
- Campus Safety (emergency): 503-370-6911

10.4 Religious Accommodations

Students requiring accommodations for religious observances should notify the instructor within the first two weeks of the semester.

10.5 Land Acknowledgement

We respectfully acknowledge that Willamette University is located on the ancestral lands of the Kalapuya people, who today are represented by the Confederated Tribes of the Grand Ronde and the Confederated Tribes of the Siletz Indians. We honor their deep connection to this land and recognize the ongoing contributions of Indigenous peoples to our academic community.