Tetris Game Development Project

Project Progress & Planning Summary

Team: Error 404: Name Not Found

Project Duration: September 8 – November 23, 2025

Total Sprints: 2 (Sprint 1: 5 weeks, Sprint 2: 5 weeks)

Table of Contents

Project Overview

Mission Statement

Develop a fully functional Tetris game demonstrating:

- Modern software development practices
- Team collaboration and agile methodology
- Comprehensive testing strategies
- Clean architecture and design patterns

Team Members

- **Anna Dinius** - Game Logic & Scoring Specialist
- **Cody King** - User Experience & Game Mechanics
- **Owen Newberry** - Rendering & Game Progression
- **Jeffrey Perdue** - Project Manager

Project Overview (Continued)

Repository & Resources

GitHub Repository:
https://github.com/jeffreyperdue/ase-420-team-project

Project Timeline:

- **Sprint 1:** September 8 – October 13, 2025 (5 weeks)
- **Sprint 2:** October 20 – November 23, 2025 (5 weeks)
- **Total Duration:** 10 weeks

Sprint 1: Planning

Sprint 1 Goals

1. Build a working MVP Tetris game
2. Write basic requirements and track completion
3. Share work on GitHub and GitHub.io
4. Establish foundation for future enhancements

Sprint 1 Scope

- **Total Features Planned:** 9
- **Total Requirements Planned:** 9
- **Focus:** Core game mechanics and basic functionality

Sprint 1: Feature Assignments

Anna Dinius - Board & Line Clearing

Features:

- Build playing field grid
- Detect and clear full lines

Deliverables:

- `board.py` - Board management
- Line clearing logic
- Unit tests for board operations

Sprint 1: Feature Assignments (Continued)

Cody King - Pieces & Collision

Features:

- Represent game pieces (type, rotation, position, color)
- Allow pieces to move and rotate
- Prevent illegal moves with collision detection

Deliverables:

- `piece.py` - Piece representation
- `figures.py` - Shape definitions
- Collision detection system

Sprint 1: Feature Assignments (Continued)

Owen Newberry - Rendering & Controls

Features:

- Render empty board on screen
- Render falling piece and locked pieces
- Map keyboard input to intents
- Display "Game Over" overlay

Deliverables:

- `pygame_renderer.py` - Visual rendering
- `input.py` - Input handling
- `app.py` - Main game loop

Sprint 1: Timeline

Week-by-Week Plan

Week 1 (9/8 - 9/14):

- Team rules and repository setup
- Project scaffolding
- Constants and figures established

Week 2 (9/15 - 9/21):

- Board class creation
- Initial Row + LinkedList approach
- Basic renderer and input mapping

Sprint 1: Timeline (Continued)

Week 3 (9/22 - 9/28):

- Line-clear implementation
- Collision/placement/movement
- Keyboard intents wired
- Expanded test coverage

Week 4 (9/29 - 10/5):

- Edge-case tests
- Error handling
- Main loop integration
- Code cleanup

Sprint 1: Timeline (Continued)

Week 5 (10/6 - 10/12):

- Final integration (branches merged)
- Documentation and comments
- Manual testing
- Sprint 1 presentation preparation

Sprint 1: Results

Completion Metrics

- **Features Completed:** 8 out of 9 (88.9%)
- **Requirements Completed:** 8 out of 9 (88.9%)
- **Team Burndown Rate:** 88.9%
- **Total Lines of Code:** 2,442 (excluding starter code)

Outstanding Item

- Game Over overlay (moved to Sprint 2)

Sprint 1: Results (Continued)

Individual Contributions

Anna Dinius:

- Lines of Code: 1,434
- Features: 2 (Board & Line Clearing)
- Requirements: 2

Cody King:

- Lines of Code: 832
- Features: 3 (Piece Representation, Movement, Collision)
- Requirements: 3

Sprint 1: Results (Continued)

Individual Contributions

Owen Newberry:

- Lines of Code: 176
- Features: 3 (Rendering, Input, Game Over)
- Requirements: 3

Note: Game Over overlay was planned but completed in Sprint 2

Sprint 1: Achievements

What Went Well

✅ Clear Separation of Concerns

- Modular architecture (Board, Piece, Game, View, Input)
- Each component had clear responsibilities

✅ High Test Coverage

- Comprehensive unit tests for Board/Row/Piece
- Core logic stable and well-tested

✅ Working Game Loop

- Functional pygame loop
- Keyboard controls responsive

Sprint 1: Challenges

What Went Wrong

⚠️ Game Over Overlay

- Not implemented by Sprint 1 deadline
- Moved to Sprint 2 for completion

⚠️ Integration Issues

- Some merging challenges between team members
- Resolved through better communication

Sprint 1: Lessons Learned

Improvements for Sprint 2

- Finish Game Over overlay early in next sprint
- Improve communication for smoother integration
- Maintain quality standards established in Sprint 1

Sprint 2: Planning

Sprint 2 Goals

1. Implement scoring system with line clear bonuses
2. Add next piece preview display
3. Implement pause/resume functionality
4. Create difficulty levels with increasing speed
5. Enhance game over screen with detailed stats
6. Add ghost piece preview

Sprint 2: Planning (Continued)

Sprint 2 Scope

- **Total Features Planned:** 6
- **Total Requirements Planned:** 35
- **Focus:** User experience enhancements and game polish

Sprint 2: Feature Assignments

Anna Dinius - Scoring & Enhanced Screens

Features:

- Scoring System
- Enhanced Game Over Screen
- Start Screen

Requirements: 14 total

- Scoring logic with multipliers
- Session high score tracking
- UI display and persistence
- Start screen with controls
- Game over screen with stats

Sprint 2: Feature Assignments (Continued)

Cody King - User Experience Features

Features:

- Next Piece Preview
- Pause/Resume Functionality

Requirements: 12 total

- Preview display area
- Piece generation integration
- Pause state management
- Input handling for pause

Sprint 2: Feature Assignments (Continued)

Owen Newberry - Game Mechanics

Features:

- Difficulty Levels
- Ghost Piece

Requirements: 9 total

- Level progression system
- Speed adjustment
- Landing position calculation
- Ghost piece rendering

Sprint 2: Timeline

Week-by-Week Plan

Week 1 (10/20 - 10/26):

- Anna: Scoring system foundation
- Cody: Next piece preview UI
- Owen: Difficulty level framework

Week 2 (10/27 - 11/2):

- Anna: Scoring enhancements with multipliers
- Cody: Pause/resume mechanics
- Owen: Ghost piece collision logic

Sprint 2: Timeline (Continued)

Week 3 (11/3 - 11/9):

- Anna: Score display & persistence
- Cody: Input handling for pause
- Owen: Difficulty progression

Week 4 (11/10 - 11/16):

- Anna: Start screen implementation
- Cody: UI polish for preview/pause
- Owen: Ghost piece rendering

Sprint 2: Timeline (Continued)

Week 5 (11/17 - 11/23):

- Anna: Game over screen implementation
- Cody & Owen: Final integration & testing
- Team: Complete documentation and final presentation

Sprint 2: Results

Completion Metrics

- **Features Completed:** 6 out of 6 (100%)
- **Requirements Completed:** 35 out of 35 (100%)
- **Team Burndown Rate:** 100%
- **Sprint 2 Status:** ✅ Complete

Sprint 2: Results (Continued)

Individual Contributions

Anna Dinius:

- Features: 3 (Scoring System, Start Screen, Game Over Screen)
- Requirements: 14

Cody King:

- Features: 2 (Next Piece Preview, Pause/Resume)
- Requirements: 12

Owen Newberry:

- Features: 2 (Difficulty Levels, Ghost Piece)
- Requirements: 9

Sprint 2: Achievements

Completed Features

✅ Scoring System

- Base scoring with multipliers
- Session high score tracking
- Level-based scoring bonuses

✅ Next Piece Preview

- Visual preview of upcoming piece
- Integrated with piece generation

✅ Pause/Resume

- Full pause functionality

Sprint 2: Achievements (Continued)

Completed Features

✅ Difficulty Levels

- Level progression system
- Increasing fall speed
- Level-based scoring multipliers

✅ Enhanced Screens

- Start screen with controls
- Enhanced game over screen
- Professional UI polish

✅ Ghost Piece

Overall Project Metrics

Total Project Statistics

- **Total Lines of Code:** 6,877
  - Source Code: ~1,200 lines
  - Test Code: ~5,677 lines
- **Total Features:** 14
  - Sprint 1: 8 features
  - Sprint 2: 6 features
- **Total Requirements:** 43
  - Sprint 1: 8 requirements
  - Sprint 2: 35 requirements

Overall Project Metrics (Continued)

Quality Metrics

- **Total Tests:** 411 test cases
- **Test Coverage:** Comprehensive across all modules
- **Test Categories:**
  - Unit Tests: 17 files
  - Integration Tests: 11 files
  - Acceptance Tests: 5 files
  - Regression Tests: 5 files

Overall Project Metrics (Continued)

Burndown Summary

- **Sprint 1 Burndown:** 88.9% (8/9 requirements)
- **Sprint 2 Burndown:** 100% (35/35 requirements)
- **Overall Project Completion:** 100% of planned features

Team Contributions Summary

Anna Dinius - Total Contribution

Sprint 1:

- Features: 2 (Board & Line Clearing)
- Requirements: 2
- Lines of Code: 1,434

Sprint 2:

- Features: 3 (Scoring, Start Screen, Game Over)
- Requirements: 14

Total:

- Features: 5

Team Contributions Summary (Continued)

Cody King - Total Contribution

Sprint 1:

- Features: 3 (Piece Representation, Movement, Collision)
- Requirements: 3
- Lines of Code: 832

Sprint 2:

- Features: 2 (Next Piece Preview, Pause/Resume)
- Requirements: 12

Total:

- Features: 5

Team Contributions Summary (Continued)

Owen Newberry - Total Contribution

Sprint 1:

- Features: 3 (Rendering, Input, Game Over)
- Requirements: 3
- Lines of Code: 286

Sprint 2:

- Features: 2 (Difficulty Levels, Ghost Piece)
- Requirements: 9

Total:

- Features: 5

Key Achievements

Technical Excellence

✅ Clean Architecture

- Layered architecture with clear separation
- SOLID principles applied throughout
- Design patterns (Factory, Singleton, Strategy)

✅ Comprehensive Testing

- 411 test cases across 4 categories
- High code coverage
- Regression prevention

Key Achievements (Continued)

Technical Excellence

✅ Code Quality

- Well-documented codebase
- Consistent coding standards
- Maintainable structure

✅ Performance

- Efficient bitmask operations
- Optimized data structures
- Smooth 60 FPS gameplay

Key Achievements (Continued)

Team Collaboration

✅ Effective Communication

- Regular progress updates
- Clear feature ownership
- Successful integration

✅ Agile Methodology

- Sprint-based development
- Regular retrospectives
- Continuous improvement

Key Achievements (Continued)

Project Delivery

✅ On-Time Delivery

- Both sprints completed on schedule
- All major milestones met
- Final presentation delivered

✅ Quality Standards

- Production-ready code
- Comprehensive documentation
- Professional presentation

Lessons Learned

What Worked Well

1. **Clear Role Definition**
    - Each team member had specific responsibilities
    - Reduced overlap and conflicts
2. **Modular Architecture**
    - Enabled parallel development
    - Made integration smoother
3. **Testing from Start**
    - Comprehensive test coverage
    - Prevented regression issues

Lessons Learned (Continued)

What Worked Well

4. **Regular Communication**
    - Weekly progress updates
    - Quick issue resolution
    - Better coordination
5. **Agile Approach**
    - Flexible planning
    - Adaptable to changes
    - Continuous improvement

Lessons Learned (Continued)

Areas for Improvement

1. **Documentation**
    - Earlier documentation updates
    - More inline comments
    - Better API documentation
2. **Communication**
    - More frequent check-ins
    - Earlier issue escalation
    - Better status visibility

Project Completion

Final Status

✅ All Features Complete

- 14 features implemented
- 100% feature completion
- All requirements met

✅ Quality Standards Met

- Comprehensive testing
- Code quality maintained
- Documentation complete

Project Completion (Continued)

Final Deliverables

✅ Working Game

- Fully functional Tetris game
- All planned features implemented
- Production-ready code

✅ Documentation

- User guide
- Architecture documentation
- Technical specifications

Project Completion (Continued)

Final Deliverables

✅ Testing Suite

- 411 test cases
- Multiple test categories
- Comprehensive coverage

✅ Presentations

- Sprint 1 retrospective
- Final project presentation
- Progress documentation

Project Completion (Continued)

Repository Status

✅ GitHub Repository

- Well-organized codebase
- Complete commit history
- Clear branch structure

✅ Documentation

- README files
- Progress reports
- Technical documentation

Project Success Factors

Key Success Elements

1. **Clear Planning**
    - Well-defined sprints
    - Specific feature assignments
    - Realistic timelines
2. **Team Collaboration**
    - Effective communication
    - Mutual support
    - Shared goals

Project Success Factors (Continued)

Key Success Elements

3. **Technical Excellence**
    - Clean architecture
    - Comprehensive testing
    - Quality code
4. **Agile Methodology**
    - Flexible approach
    - Continuous improvement
    - Regular retrospectives

Project Impact

Technical Impact

- **Architecture:** Scalable, maintainable design
- **Testing:** Comprehensive test coverage
- **Code Quality:** Production-ready standards
- **Documentation:** Complete technical specs

Project Impact (Continued)

Team Impact

- **Skills Development:** Enhanced technical skills
- **Collaboration:** Improved team coordination
- **Process:** Established agile practices
- **Experience:** Real-world project experience

Project Impact (Continued)

Educational Impact

- **Best Practices:** Applied software engineering principles
- **Design Patterns:** Implemented multiple patterns
- **Testing:** Comprehensive test strategies
- **Documentation:** Professional documentation standards

Conclusion

Project Summary

✅ Successfully Completed

- All planned features implemented
- All requirements met
- High quality standards maintained

✅ Team Achievement

- Effective collaboration
- On-time delivery
- Professional results

Conclusion (Continued)

Key Takeaways

- **Planning Matters:** Clear planning enabled success
- **Communication is Key:** Regular updates prevented issues
- **Quality First:** Testing and documentation paid off
- **Agile Works:** Flexible approach handled changes well

Thank You

Project Completion

Tetris Game Development Project
Error 404: Name Not Found

Status: ✅ Complete
Quality: ✅ Excellent
Delivery: ✅ On Time

Thank you for following our progress!

End of Presentation

For more information:

- GitHub: https://github.com/jeffreyperdue/ase-420-team-project
- User Guide: `docs/user_guide.marp.md`
- Architecture: `docs/design_architecture.marp.md`
- Final Presentation: `docs/final_presentation.marp.md`