# Introduction

This document outlines the requirements for the Technical Design Document (TDD) required for GAM150. All listed sections must be included, and they must be in the order listed.

While this document is due at the engine proof milestone, you should start working on it immediately when your GDD is finished (if not before). This is the document that will tell every programmer on your team what they need to know to really start implementing the game. A really good TDD will essentially just describe what you have already done for your engine proof. Most student games need only a 15-25 page TDD. ***However, 7-15 pages are recommended for GAM150 projects.***

While you should try to think through as much of your technical design as possible in this document, don’t worry about getting it perfect or entirely fleshed out. In particular, you should cut the design down to what you need for a really good first playable. Anything in the design that isn’t critical to a good first playable should be labeled as such (make these things a different color, put them in italics, or whatever works for you).

Some overall guidelines:

* Every page must have a page number (except for the cover page), your game title, and the DigiPen copyright (“All content © 2014 DigiPen (USA) Corporation, all rights reserved.”).
* Documents must be neatly formatted and easily readable. Put page breaks before new sections (when appropriate), use consistent formatting and fonts, use headings for sections and sub-sections, etc. Sloppy documents are unprofessional and may prevent your project from getting funding.
* Spell-check and grammar-check the document before submission.
* Avoid the use of personal pronouns (I, we, etc.)

# Cover Page

The cover page should contain the following information:

* Game Title
* “Technical Design Document”
* Class name and section (e.g. GAM150S14-A – GAM150S14-E)
* Semester and year (e.g. Spring 2014)
* Team Name
* Team Roster – List all members of the team, including the following information:
  + Student name
  + Official job (or jobs)
  + Coding responsibilities

# Table of Contents

The TDD must contain a table of contents (TOC). Make sure the TOC is updated every time the TDD is submitted. If necessary, refer to the Word documentation for help on adding and updating TOCs.

**TDD Structure**

## Overview

Describe the overall architecture of the project. List and describe each global component the project uses (graphics, physics, AI, UI, file I/O, memory manager, audio, etc.).

## Graphics Implementation

What techniques will be used to implement the graphics design? How are you loading assets such as backgrounds, sprites, and animations?

## Multiplayer Implementation

What techniques will be used to implement multiplayer? For single-player games this section can be omitted or contain a sentence explicitly stating that the game is single player.

## Coding Methods

Describe all coding conventions that are to be used on the project, including file naming conventions, file locations (i.e., which files go in which folders in the project), code formatting, and code documentation. Also describe what kind of source control system you will use and any rules your team has about its use.

## Debugging

Describe the support you have for debugging in your game. Do you have an in-game debug console (or at least a simple output-only console)? Do you have a debug drawing system? Do you have a clean assertion system? Do you have an in-game performance viewer? Do you have a way to watch variables in-game? Can you shut off your debug controls?

## Tools

Describe any third-party tools used to develop the game. This includes the C compiler (GNU, Visual Studio), map editors, art and audio processing tools, testing tools, etc.

Describe any custom tools that will be developed for the game, including content editors and conversion tools.

## Scripting Languages

Describe any scripting languages that you have created for the game. No third-party scripting languages may be used in a GAM150 project.

## Technical Risks

Describe the top two to three technical risks to the project; such as features that might be overly complex, poorly understood, or potentially beyond the capabilities of the platform without additional work. For each technical risk include a plan for mitigating the risk.

# Appendices

## Appendix A: Interface Flow

### Flowchart

Create a flowchart that illustrates the entire flow of the interface that is not related to actual gameplay. This should include your intro screen, the main menu, pause screen, setup screens, character or level selection screens, cut-scenes, victory screens, networking screens, etc.

### Mockups

Create a mockup and description for each of the screens listed above (this can be very rough—hand drawn even). If this seems like too much work, you should probably have a simpler interface and/or a simpler game.

## Appendix B: Art Requirements

Describe the requirements for all art assets used in the game, including file naming conventions, file format(s), and anything else needed to create the assets.

Describe the source or sources of all the art in your game (programmer art, DigiPen libraries, BFA students, etc.). Remember that you must create all your own art (or use DigiPen libraries). You cannot use your friends, family members, public domain material, etc. You can use art from other students not on your team (but you must give them credit). If you wish to use DigiPen artists, you must talk to your instructor first. You can never use outside artists at all.

## Appendix C: Audio Requirements

Describe the requirements for all audio assets used in the game, including file naming conventions, file format(s), and anything else needed to create the assets.

Describe the source or sources of all the audio in your game (programmer audio, DigiPen libraries, other students, etc.). Remember that you must create all your own audio (or use DigiPen libraries). You cannot use your friends, family members, public domain material, etc. You can use audio from other students not on your team (but you must give them credit). You can never use outside musicians at all.