[Project Title]

[Project Subtitle]

Joe Student
foo@email.com
Jane Student
foo@email.com
University of Nebraska—Lincoln

Fall 2525
Version 1.x

[Provide a short abstract of this document here. Throughout this template we give directions and *placeholders* within square brackets on what the content should be. As you update this document, the placeholders should be removed.]

Revision History

Version	Description of Change(s)	Author(s)	Date
1.0	Initial draft of this design document	Joe Student	2525/01/01
1.1	Typo and grammar fixes	Jane Student	2525/01/02

Contents

1 Introduction

[Provide a short introduction to this document, the project and the context in which it is being developed. This document needs to conform to the IEEE 1016 standard [1] (this is how you use citations in LaTeX and BibTex). Do not include citations just to include them. Only include them if they are relevant *and* you have cited them somewhere in the document.

Keep your writing professional and write in a *technical style*. Keep things short but provide sufficient details so that another development team would be able to reproduce your system. Do not use personal pronouns. Write in a present tense *as if* the project is already done. This document is about the *design* of the system not about the process. Do not refer to "we did A then we did B, etc." Do not write as if this is a class project; write *as if* it were a real project.

The introduction should include details on who the client is, what is their business model, what are the major *features* (not components) that the system needs to have and support? When you describe the components in Sections 2 and 3, you should justify your design decisions by calling back to these requirements as necessary. If you don't include these details, there is no way for you to justify your design decisions.]

1.1 Purpose of this Document

[Describe the purpose of this document; the goal(s) that its content are intended to achieve]

1.2 Scope of the Project

[Describe the scope of the project, what features and functionality it covers (at a high-level). What does this project cover and what does the project *not* cover (ie what is the responsibility of another team?).]

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

[Define any terms that require a definition–domain specific terms, non-standard terms, or terms that are used in non-standard ways. Write for a *technical audience*: do not include things that are part of the course or that a technical audience would know already.]

1.3.2 Abbreviations & Acronyms

[Define all abbreviations and acronyms used in this document here. Only include them if you actually use them in the document. When used in the document for the first time, it should include the full text. For example, the Associate for Computing Machinery (ACM) is a global organization of computing professionals and researchers that promotes the science and profession of computing.]

ACM Association for Computing Machinery

IEEE Institute of Electrical and Electronics Engineers

UAV Unmanned Aerial Vehicle

2 Overall Design Description

[Provide an overall *high-level* and abstract description of the design. Identify the major components, major features, etc. Identify the technologies used Identify and talk about the *major* features of the project; this section should be updated throughout the drafts]

2.1 Alternative Design Options

[If applicable, describe and discuss alternative design options that you considered and discuss why they were not chosen. What advantages and disadvantages do the alternatives provide and what advantage/disadvantages do the chosen design elements provide. Provide some justification for why the chosen elements? advantages/disadvantages outweighed the alternatives]

3 Detailed Component Description

[Provide an introduction to this section here. Identify the next subsections and what each one will cover.]

3.1 Database Design

[This section will be used to detail your database schema design (Phase III). For your draft, you should provide a sketch of your ER diagram (which may be replaced with a more formal one after your implementation. Identify all tables and their purpose; the columns in each table, etc. Do not include a lot of text; your ER diagram should provide enough technical details. Your text should not be redundant to your diagram. Instead, your text should *justify* the design; make reference to the introduction and to the client's business model.]

3.1.1 Component Testing Strategy

[This section will describe your approach to testing this particular component. Describe any test cases, unit tests, or other testing components or artifacts that you developed for this component. How was test data generated (if a tool was used, this is a good opportunity for a citation). How many test cases did you have; how many of each type? *Justify* why that is sufficient. What were the outcomes of the tests? Did the outcomes affect development or force a redesign?

You may refer to the course grader system as an external testing environment "provided by the client" or "another QA/testing team".]

3.2 Class/Entity Model

[This section should detail your classes–their state, interface and how they relate to each other. Your draft should include a sketch (hand or tool generated) of your classes using a UML diagram. Figures and tables should have proper captions and be referenced in the main text just like in Figure 1. Don't have one giant UML diagram; break it up into subfigures, collecting related classes as appropriate. Your draft needs to provide enough detail that we can give feedback on your design before you submit the code for each phase. Your sketches should be replaced with formal diagrams in later drafts.

Identify which classes are responsible for each feature. Classes should follow the *Single Responsibility Principle.* This section should be updated throughout each phase as you add more classes.]

3.2.1 Component Testing Strategy

[This section will describe your approach to testing this particular component. Describe any test cases, unit tests, or other testing components or artifacts that you developed for this component. How was test data generated (if a tool was used, this is a good opportunity for a citation). How many test cases did you have; how many of each type? *Justify* why that is sufficient. What were the outcomes of the tests? Did the outcomes affect development or force a redesign?

You may refer to the course grader system as an external testing environment "provided by the client" or "another QA/testing team".]

Figure 1: A UAV (Unmanned Aerial Vehicle) soars above Memorial Stadium. Figures should be numbered and properly captioned. This is just an example of how to properly include a figure.

3.3 Database Interface

[This section will be used to detail phase IV where you modify your application to read from a database rather than from flat files. This section will detail the API that you designed–how it conformed to the requirements, how it worked, other tools or methods that you designed to assist, how it handles corner cases and the expectations or restrictions that you've placed on the user of the API. What is "good" data and what is considered "bad" data and how does your API handle it? An example table is presented as Table 1.]

Table 1: Average Performance on Assignments; on-time vs. late and individual vs partners. In general, captions for Tables should appear above the table.

	1	2	3	4	5	6	7
On-time	93.16% (78.46%)	88.06% (72.31%)	87.89% (67.69%)	89.37% (56.92%)	83.42% (29.23%)	88.40% (53.85%)	74.56% (75.38%)
Late	88.75% (12.31%)	85.28% (20.00%)	70.32% (15.38%)	90.40% (15.38%)	82.74% (44.62%)	94.22% (15.38%)	N/A
Diff	4.42%	2.79%	17.57%	1.03%	0.68%	5.82%	-
Individual	NA	88.43% (73.85%)	82.32% (33.85%)	87.22% (27.69%)	86.40% (23.08%)	82.67% (26.15%)	
Pairs	NA	83.55% (18.46%)	86.22% (49.23%)	91.00% (46.15%)	78.53% (49.23%)	92.83% (46.15%)	
Diff	NA	4.88%	3.90%	3.78%	7.87%	10.16%	

3.3.1 Component Testing Strategy

[This section will describe your approach to testing this particular component. Describe any test cases, unit tests, or other testing components or artifacts that you developed for this component. How was test data generated (if a tool was used, this is a good opportunity for a citation). How many test cases did you have; how many of each type? *Justify* why that is sufficient. What were the outcomes of the tests? Did the outcomes affect development or force a redesign?

You may refer to the course grader system as an external testing environment "provided by the client" or "another QA/testing team".]

3.4 Design & Integration of a Sorted List Data Structure

[This section will be used to detail phase V where you design and implement a custom data structure and integrate it into your application. Is your list node based or array based? What is its *interface* and how does it define a sorted list? Is it generic? Why? You can/should provide another UML diagram for this list.

You should not include large chunks of code in your document, but if you do need to typeset code in LaTeX the recommendation is the `minted` package. You can typeset code inline like this: `List<Integer> myList` or you can define entire blocks (usually placed within a figure environment) like in Figure 2.

```java
/**
 * A basic Hello World class
 */
public class HelloWorld {

  public static void main(String args[]) {
    System.out.println("Hello World");
  }
}
```

Figure 2: A basic Hello World program in Java.

If you do use the minted package, you may need a substantial amount of additional setup. First, you need to install the LaTeX `minted` package installed *and* you need to install the python pygments package (see `https://pygments.org/download/`; you can use `pip install pygments`). Once everything is installed, you must compile using the `--shell-escape` flag.]

3.4.1 Component Testing Strategy

[This section will describe your approach to testing this particular component. Describe any test cases, unit tests, or other testing components or artifacts that you developed for this component. How was test data generated (if a tool was used, this is a good opportunity for a citation). How many test cases did you have; how many of each type? *Justify* why that is sufficient. What were the outcomes of the tests? Did the outcomes affect development or force a redesign?

You may refer to the course grader system as an external testing environment "provided by the client" or "another QA/testing team".]

4 Changes & Refactoring

[During the development lifecycle, designs and implementations may need to change to respond to new requirements, fix bugs or other issues, or to improve earlier poor or ill-fitted designs. Over the course of this project such changes and refactoring of implementations (to make them more efficient, more convenient, etc.) should be documented in this section. If not applicable, this section may be omitted or kept as a placeholder with a short note indicating that no major changes or refactoring have been made.]

5 Additional Material

[This is an optional section in which you may place other materials that do not necessarily fit within the organization of the other sections.]

References

[1] IEEE standard for information technology–systems design–software design descriptions. *IEEE STD 1016-2009*, pages 1–35, July 2009.

[2] Bruce Eckel. *Thinking in Java*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 4th edition, 2005.

[3] Internet Goons. Do I cast the result of malloc? http://stackoverflow.com/questions/605845/do-i-cast-the-result-of-malloc. [Online; accessed September 27, 2015].