The project design is a part of the Sprint document that is delivered with each increment and represents a series of software design artifacts, models, and techniques, accompanied by associated written documentation, that fully describes the software design as a lead in to the development and testing stages. The project design should include the following parts:

1. **Software Design Patterns (SDPs):** Software design patterns typically show relationships and interactions between classes or objects at a conceptual level, for domain independent structure and behavior. A pattern provides a body of knowledge on a particular structure thereby communicating insight on a portion or component of a solution; the idea is that we can leverage patterns that are generalized from prior solutions and experience to build solutions more effectively. Each project will need to identify, define, explain, and document a set of one or more SDPs including: High-level Patterns such as Model View Controller or MVC with Observer; Creational Patterns such as Abstract Factory, Builder, etc.; Structural Patterns such as Adaptors, Proxy, etc; Behavioral Patterns such as Interpreter, Mediator, etc.; and Concurrency Patterns such as Event-Base Asynchronous, Scheduler, etc.

2. **Detailed Design:** This includes software and database design. For software design, you are to employ a wide range of UML diagrams as appropriate for your particular project and its domain requirements. Structural UML diagrams include: Class, Component, Object, Profile, Composite Structure, Deployment, and Package. Behavioral UML diagrams include: Activity, State Machine, Sequence, Communication, Interaction, and Timing. For database design, you are required to construct an entity-relationship diagram (ERD) that for the conceptual structure of the data and its interrelations to be used to eventually generate a relational database schema. Note that not all projects may have all of these diagrams – but each project should be able to select at least enough different software and database diagrams so that each team member is responsible for one aspect of the software design as realized by a respective diagram. Some details on different diagrams are:
   - An Entity-Relationship (ER) diagram for shows the conceptual structure of the data and its interrelations (see Ghezzi, section 5.6.1, or a database management textbook). You must use the ER specification capabilities of a UML tool (Eclipse plug in or Visio) in constructing your diagram. Make sure that you describe the attributes of each entity including types and acceptable ranges. For relationships, make sure you identify their directions/cardinalities.
   - UML Sequence Diagrams to show the sequential flow between your various system objects and components.
   - UML Statechart Diagrams to show the general control structure of the system and identify each state and the transitions between states.
   - UML Class/Object Diagrams that include both public, private, and protected methods, and private and protected attributes. Organize your classes into packages as appropriate, and use the various UML references and relationships to indicate which classes and/or packages collaborate with one another. You may divide the packages and classes between team members, with each team member responsible for a set of packages and classes if this aspect of your design is significant in size.
- UML Activity Diagrams (ADs) to show structural components and flow between the various states that support both forms and joins to indicate actions that can occur in parallel.

3. **Combined Design Document:** The SDPs, ERD, and UML diagrams (design artifacts) for a project must be assembled into an overall design must be coherently organized into numbered sections with all diagrams as labeled figures that are explained and referenced within the text of the design. This will include a section that brings the design together by discussion the relationship and dependencies between all of the design artifacts.

**Important Notes:**
- A team must have at least one SDP and a subset of the ERD and UML diagrams.
- Each team member is responsible for one of these diagrams.
- Each SPD, ERD, and UML diagram must be accompanied with a textual explanation. The rule of thumb is ½ page of description per ½ page of diagram.
- Over the course of the entire semester, new versions (using github as a repository) of these various design artifacts must be added to the Sprint document as you make changes or additions based on your sprint backlog and/or newly defined requirements.
- For UML, there are a myriad of sources online including: [http://www.uml.org/](http://www.uml.org/). An image search on your favorite search engine with “UML Diagrams” will show lots of samples.