1. **Course number and name**
   CptS 487/587: Software Design and Architecture

2. **Credits and contact hours**
   3 credits, 3 lecture hours

3. **Instructor's or course coordinator's name**
   Bolong Zeng

4. **Textbook, title, author, and year**
   L. Bass, P. Clements, and R. Kazman. 2012. *Software Architecture in Practice* (3rd ed.). Addison-Wesley. ISBN: 9780321815736 (Required)
   R.C. Martin. 2002. *Agile Software Development: Principles, Patterns and Practices* (1st ed.). Pearson. ISBN: 9780135974445. (Required)
   E. Gamma, R. Helm, R. Johnson, and J. Vlissides. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software* (1st ed.). Addison-Wesley. ISBN: 9780201633610. (Recommended)
   P. Clements, *et al.* 2010. *Documenting Software Architectures: Views and Beyond* (2nd ed.). Pearson Education. ISBN: 9780321552686. (Recommended)

5. **Specific course information**
   a. *Catalog description:* Software design; design principles, patterns, and anti-patterns; design quality attributes and evaluation; architectural styles, architectural patterns and anti-patterns.
   b. *Prerequisites or corequisites:* CptS 321 with a C or better; CptS 322 with a C or better; Certified major in CptS, CE, EE, or SE.

6. **Specific goals for the course**
   By the end of the course, students will be able to:
   - Explain key concepts in software design and construct professional design documents. (3a, 3b, 3c, 3e)
   - Explain and apply software design principles. (1a, 1b, 1c, 1d, 1e, 2a, 2b, 6a, 7f, 7g)
   - Design a software system to account for key issues such as concurrency, security, and data persistence. (1b, 1d, 1e, 2b, 2c, 2d, 6a, 7f, 7g)
   - Identify opportunities to apply common design patterns and identify poor design decisions and propose alternative solutions. (1a, 1c, 6a, 6c)
   - Critique a proposed software design in terms of quality attributes; select and apply techniques to evaluate the quality of a software design. (1b, 1e, 2c, 6b, 6c)
   - Describe the main software architectural styles and select the appropriate style for a given software system. (1a, 1b, 1c, 1d, 1e, 7f, 7g)
   - Identify reusable components for the system to be developed.
   - Describe common architectural patterns and apply them when appropriate. (1a, 1b, 1c, 1d, 1e, 7f, 7g)

- Produce architectural diagrams representing the various views of the system. (3a, 3e)

7. **Brief list of topics to be covered**
   - Software design concepts and documentation
   - Software design principles
   - Key design issues
   - Design patterns and anti-patterns
   - Software design quality
   - Software architectural styles, patterns, anti-patterns
   - Software reuse