# Complete class diagram

# FR1: Load data from file

* 1. **Name** the design pattern you chose.
  2. Adapter pattern
  3. Explain **why** you applied that particular design pattern (one paragraph).
  4. The adapter pattern allows the easy access of multiple types of file using a single interface. This works well for loading files as it allows me to load patient data not just from .txt file put from any file with data in the correct format using a single interface class.
  5. **Show** a class diagram that highlights the specific classes in the system and relationships that address that design pattern.
  6. Explain, step by step, **how** your design works. A numbered list may work best here. Reference your diagrams as necessary.
  7. Reference the git commits by commit ID where you addressed this aspect of the design.

# FR2: Load data from database

* 1. **Name** the design pattern you chose.
  2. Composite pattern
  3. Explain **why** you applied that particular design pattern (one paragraph).
  4. The composite pattern helps me to treat different objects as the same which works well in this case because I am loading data from files and databases but I don’t care how the data is loaded I am just looking for the data extracted. By applying the composite pattern I can treat patient data from file and patient data from a database as just patient data.
  5. **Show** a class diagram that highlights the specific classes in the system and relationships that address that design pattern.
  6. Explain, step by step, **how** your design works. A numbered list may work best here. Reference your diagrams as necessary.
  7. Reference the git commits by commit ID where you addressed this aspect of the design.

# FR3: Generate patient alerts

* 1. **Name** the design pattern you chose.
  2. Strategy pattern
  3. Explain **why** you applied that particular design pattern (one paragraph).
  4. I have to apply a strategy to generate alerts for different patients based on a long list of criteria. The strategy pattern allows me to abstract away the implementation for different diseases making it easy to add multiple diseases, and be able to define different alert criteria for each disease.
  5. **Show** a class diagram that highlights the specific classes in the system and relationships that address that design pattern.
  6. Explain, step by step, **how** your design works. A numbered list may work best here. Reference your diagrams as necessary.
  7. Reference the git commits by commit ID where you addressed this aspect of the design.

# FR4: Alert doctors

* 1. **Name** the design pattern you chose.
  2. Observer pattern
  3. Explain **why** you applied that particular design pattern (one paragraph).
  4. The observer pattern works well for notification systems because the patient doesn’t care what it’s vitals are but their doctor/s might. But we don’t know which doctors care, the observer pattern allows us to setup the system so that the doctors assign which patients vitals they care about and subscribe to their vital alerts.
  5. **Show** a class diagram that highlights the specific classes in the system and relationships that address that design pattern.
  6. Explain, step by step, **how** your design works. A numbered list may work best here. Reference your diagrams as necessary.
  7. Reference the git commits by commit ID where you addressed this aspect of the design.