This assignment is to be done individually. For the two case studies below, identify and explain potential professional, ethical, legal, security, and social issues and responsibilities relevant to software engineering. It is not enough to just list issues/responsibilities. You must describe how the issue/responsibility is exemplified in the case study and explain the significance (in terms of potential moral harms) for the case. In addition, when appropriate, link issues to the course work and readings, including the ACM/IEEE Software Engineering Code of Ethics.

Your answers must be typed using 11pt Times New Roman font with 1 inch margins, and consider all professional, ethical, legal, security, and social issues and responsibilities. Your work will be assessed based on the following:

(a). Identification of professional, ethical, legal, security, and social issues and responsibilities
(b). Description of how the issues are exemplified in the case study
(c). Explanation of the significance of each issue in terms of potential moral harms
(d). Links to readings, course notes, and the software engineering code of ethics
(e). Overall quality of, depth of, and thought put into your analysis
(f). Quality of the writing

Be sure to include your name and date on your paper and hand it in during class on the due date.

Case Study 1.

A sales manager at LifeDesign, a medical systems engineering firm, receives a request to develop a radiation-delivery system for use in outpatient hospital settings. The request specifies that the product must have a guaranteed failsafe mechanism to prevent radiation overdoses due to user input error. The sales manager reviews the request with his development team, who tell him that (a) the system cannot be built to specifications within the desired timeframe and (b) there is no way in this kind of system to build in a guaranteed failsafe against operator error—proper radiation dosages vary so much by patient, there is no way for the system to enforce safe limits by itself—and so it must rely on accurate user input of the dosage instructions on the prescription. The best we can offer, the team tells him, is a warning system that will loudly prompt the user to verify and double-verify the correct input. But, the team tells him, it is well-known that such warnings are often mindlessly canceled by users who find them annoying, so this mechanism is hardly guaranteed to prevent operator error. The sales manager goes back to the client with a proposal that promises delivery of all functionality within the specified time frame.

When they receive the proposal, which has been accepted and contracted with the client, the engineering team begins to design the system. But it is three weeks into the work before anyone notices the delivery date—the design team leader immediately appeals to the sales manager, reminding him that he was told this was an impossible date to meet. He tells her that this is unfortunate but that they are under contract now, and
there will be severe financial penalties for a late delivery. He reminds her that upper management will not be happy with any of them in that case, and tells her that her team had better find a way to get it done. The team leader goes back to her group, frustrated but resigned, and tells them to get it done somehow. No one has even noticed yet that the contract also includes language about the guaranteed failsafe against bad user input.

Halfway through the delivery schedule, the sales manager and design team leader sit down with the client for an update. Despite being well behind schedule, they put on a good show for the client, who leaves confident that all is as it should be and that the product will deliver full functionality on time. Weeks before the due date, the team leader is telling her team that their jobs depend on an on-time delivery. Her engineering team has resorted to desperate measures—especially the group responsible for the system software, who are taking shortcuts, hiding system errors, doing sloppy coding and subcontracting work out to third-parties without proper review of their qualifications. When the product is finally assembled, there is no time to test the full system in the actual client/user environment, so the quality-assurance team of LifeDesign relies on computer simulations to test its operation. These simulations make many flawed assumptions due to the rush schedule and a lack of information about the user context/environment. None of the simulations assume erroneous dosage input by users.

The product is delivered on time, and represented as having full functionality. As a result, the client tells its therapists using the new system that it has an advanced failsafe mechanism that will prevent radiation overdoses from user error.

**Case Study 2.**

In 2007, Google launched its StreetView feature, which displays searchable aerial and street-view photographs of neighborhoods, city blocks, stores, and even individual residences. From the very beginning privacy concerns with Google’s technology were evident; it did not take long for people to realize that the feature displayed photographs of unwitting customers leaving adult bookstores and patients leaving abortion clinics, children playing naked, adults sunning themselves topless in their backyards, and employees playing hooky from work. Moreover, it was recognized that the display of these photos was being used by burglars and other criminals to identify ideal targets. Although Google did initially think to remove photos of some sensitive locations, such as domestic violence shelters, it was initially very difficult for users to request removal of photos that compromised their privacy. After an initial outpouring of complaints and media stories on its privacy problems, Google streamlined the user process for requesting image removal. This still presupposed, however, that users were aware of the breach of their privacy.

In 2010, StreetView became the center of a new privacy scandal; it was discovered that software used in Google vehicles doing drive-by photography had been collecting personal data from unencrypted Wi-Fi networks, including SSIDs (user-assigned network ID names), device identifiers, medical and financial records, passwords and email content. Initially Google claimed that this data had not been collected, later they said that only ‘fragments’ of such data had been retained, yet eventually they conceded that
complete sets of data had not only been collected but stored. At one point Google blamed the breaches on a single ‘rogue engineer,’ though later it was learned that he had communicated with his superiors about the Wi-Fi data collection.