4.1:

1. System Requirements. Should define exactly what is to be implemented.
2. System Requirements. Should define exactly what is to be implemented.
3. User Requirements. What features the user wants, and what functionality they expect.
4. Functional Requirements. States what the system should or should not do.
5. Non-Functional Requirements. The timing constraints, development process, regulation, and other constraints.

4.2

- What types of tickets are offered? Two-way? one-way? How long are these tickets valid for?

4.3

- An automated ticket machine sells rail tickets. The user uses menus to select a destination, and ticket type. The user is then asked to input their credit card and pin. The machine then charges the card, and prints the selected ticket.

4.4

- The system must be done in one year. There are also city government regulations on ticket systems. There are also hardware constraints due to the type of ticketing machine used.

4.6

- By using shared web documents, spreadsheets, and databases. To help keep organized the relationships between the requirements.

4.7

- A person wants to check their account balance.
- A person wants to withdraw a $20 bill.
- A person wants to withdraw $200 dollars all in $10 bills.

4.8

- Anyone who is involved in developing and maintaining the requirements, relevant developers, and stakeholders to ensure the correct requirements have been identified, and any questions clarified.

4.10

- Give reasons why particular requirements might be ambiguous. Communicate with people to get these ambiguities resolved. Work to improve the current employers requirements, rather that changing it to match your previous employers.