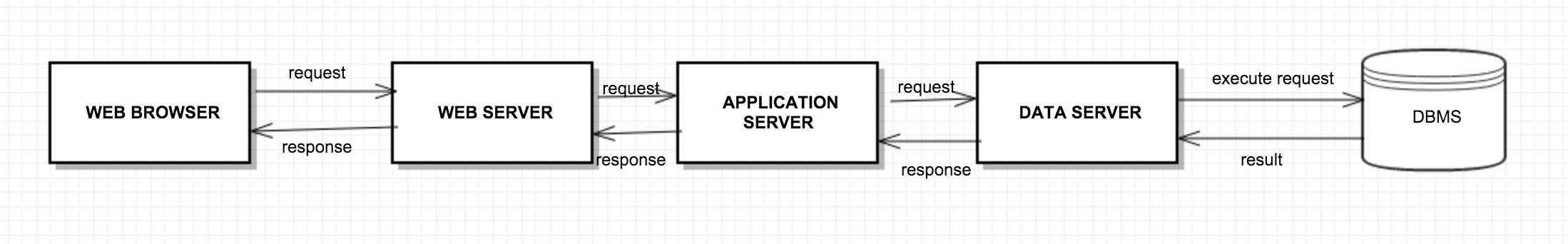
1. **Software Requirement Specification:**

SRS is documented in a separate file. Please refer ‘SRS\_ULTRON.doc’.

**2. Architecture** The architecture that we are using for our project E-healthcare is client-server architecture. This architecture best suits our requirements, as the project we are building is a web application. We are using this architecture because in this design we can make changes very easily to the system. This architecture supports the interactive process between the client and server, which is very important in our system because our system is mainly used as mode of communication between the doctors and patients.



**3. Diagrams -** These diagrams are in folder 'UML'

**Use Case Diagram**

**Class Diagram**

**Sequence Diagram**

**4. Test plan**

Please refer the document ‘test\_plan\_ultron.doc’

**5. Updated Risk management**

New risks identified are listed in points 11, 12 and 13 under RISK IDENTIFIED, during the progress of project from requirement analysis stage to development stage. Corresponding plan to mitigate these risks are listed in points 11, 12 and 13 under PLAN FOR MONITORING RISKS.

**A. Risks identified:**

1. Technical limitations in implementing some requirements will force to change those requirements.
2. Unrealistic schedule, as we don’t have prior knowledge of working on similar type of projects.
3. Unfamiliar areas of project take more time than expected to design and implement.
4. The features requested might be beyond what the development team can deliver in the time available.
5. Specific areas of the product may consume more time than expected.
6. Quality may get affected because of the upcoming deadlines.
7. If documentation consumes considerable time, project may cross the deadline.
8. Conflicts among team members result in poor communication and may end up critical development work being performed by few developers.
9. Everyone doesn’t have same level of expertise across all the technologies.
10. Lack of industry level experience in the team.
11. GUI design may be too complex for non-technical users.
12. Estimating correctly number of users for the final product basing on which database design and efficient working of GUI is dependent on. Like if number of users,expectation is more then, one should include indexing and various other mechanisms to increase performance of DB queries.
13. Costs associated with defective product.

**B. Plan for monitoring risks:**

1. Plan ahead and do enough research regarding the stream we are working on.

2. Arrange frequent meetings and reviews and make use of Mom’s.

3. A) Hire new resources.

B) Push the requirement to the next build / iteration to develop.

C) Over time – This may affect quality of the product.

Note: As options A & B won’t be possible in our team. So, we left with only option C.

4. Plan ahead and build expertise in technical as well as functional knowledge.

5. Proper scheduling helps us avoiding pitfalls and makes the product development easy without bottlenecks.

6. Do documentation in parallel to development.

7. Equal distribution of work might reduce conflicts. In addition to that, arranging team hangouts, frequent get to together and stress relieving activities are some of the helpful factors in reducing team conflicts.

8. Knowledge transfers and helping each other in their tasks will bring everyone on the same pace.

9. Upgrade you self and put more man hours to get sufficient expertise.

10. Good knowledge on various software technologies and tools and their limitations ahead of starting the project will help overcome requirement changes due to technical difficulties.

11. Users will be provided with online help guide to assist them regarding the fine details involved with E-Healthcare system.

12.Better to develop DB with good capacity and tune it with best performance parameters because health care applications always involve huge datasets.

13. Frequent incremental releases along with constant monitoring from client and project leader’s side decreases the costs invoked because of defective product.

**Note:** Risks identified in section A and plan to handle those in B are mapped in order.

**C. Contingency plans for these risks:**

1. Considering and implementing modularity in the product development will guarantee Minimum viable product.
2. Maintaining less dependencies among interfaces will avoid delay in milestone will not steep in to other milestones.
3. Implementing best modeling and documentation techniques like ‘Agile’ model to make a project best fit in a fast changing environment.

These are some of the principles in software life cycle to handle emergencies.

**6. Updated Project plan**

The updated project plan during the progress of project from requirement analysis phase to development phase along with future testing and deployment phase is detailed in document ‘Updated\_project\_plan.docx’.

**7. Meeting Minutes**

Minutes of meeting document is listed in document ‘Minutes\_Of\_Meeting.xl’.

**8. Project summary**

Detailed project summary of historical events along with current status of our project is detailed in document ‘Project\_Summary.txt.