1. **Introduction**
   1. **Existing System:**

The existing system allows you to apply for admission only once without login, which makes the database bulky. The existing system does not have a user profile, thus user cannot login to see the details of admission. The existing system is not “mobile-friendly”, meaning, it is not responsive. The existing system allows administrator to print reports.

The workflow of the current system is such that, a user visits to apply for admission.

The user applies for the admission, without any login, that means the admission is open, that means it is open for hacking attacks, which is an issue of security.

After applying for the admission, the user is asked to pay the admission fee, this is done through a secure payment gateway.

This portal is for all Bachelor and Master Courses, except for B.E.

For admission of B.E there are different criteria, and to fulfill that there is another different web portal.

This portal takes different parameters as those parameters require for B.E admission process.

By adding a different portal is plausible that there is another **database** and by that extension, more tables in that database, which creates unnecessary load on server.

There is no centralized login for the faculties or the other necessary departments.

* 1. **Problem Definition:**

**T**he current system has many problems such as; it does not have an integrated quiz for admission. It is not mobile friendly. It does not have admin panel, authorities have to ask system administrator to make changes. It does not have a good user interface.

As per our preliminary research, the current system for the admission process for the university is open for all, due to which the University cannot keep a track of students who have visited the admission page and did not applied for admission. The new system targets user in such a manner that, the student (user) will have to first register an account and based on which the student (user) will be able to apply for admission in different courses.

The second problem in the system is that, there is lack of authentication for each module; anyone can directly access any module, which risky as per the University point of view. So in order to avoid this problem, the secondary target will be to add authentication as security.

Another problem in the current system is that it is not ‘mobile-friendly’ aka ‘responsive’, so the aim will be to re-design the current system such that it can be accessible from any device.

**1.3 Proposed System:**

We as a team have proposed a system which overcomes most of the problems in the existing system. By definition, University Admission system will provide facility to various kinds of users of this system respectively. For students, this system will help in such a manner that, a student will be able to view course details of the respective course, which will include, course duration, course syllabus, university details, along with these facilities, a student will be able to register for certain courses.

After registration, a student will be able to apply for his/her choice of course. A student then also can have the option to pay the one-time registration fees online.

For, Administrative department, this system is helpful for tracking admission for their respective department and according to that they will be able to generate reports.

For, Accounts department, this system is helpful for tracking the payment status of the students who have registered for the courses.

Moreover, this system will have a Super Admin who will be able to manage all the respective admin, meaning Super Admin will be able manipulate the roles of admins such as Add Admin and Delete Admin .

There will be centralized login for the students and the admin committee (department).

This system overcomes limitation of two different portal i.e. one for all bachelor and master courses and another for B.E courses.

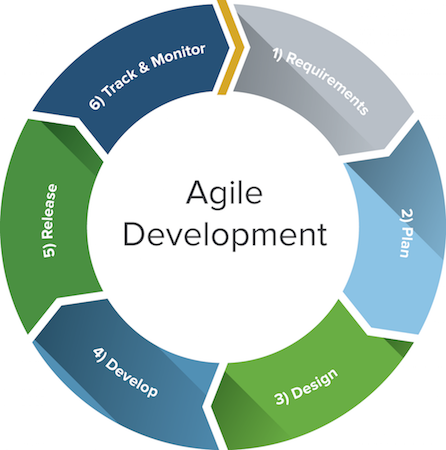
This system saves the loads on the server as it uses only **one** database.

1. **Software Development Methodology**

We have developed this project by following the **AGILE** methodology.

As software development is a **continuous** process, AGILE methodology is the best option to select among all Software Development Technology.

Below is how AGILE methodology works: -



* 1. **What is Agile methodology/?**

Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams.

Agile methods or Agile processes generally promote a disciplined project management process that encourages frequent inspection and adaptation, a leadership philosophy that encourages teamwork, self-organization and accountability, a set of engineering best practices intended to allow for rapid delivery of high-quality software, and a business approach that aligns development with customer needs and company goals.

Agile development refers to any development process that is aligned with the concepts of the Agile Manifesto.

The Manifesto was developed by a group fourteen leading figures in the software industry, and reflects their experience of what approaches do and do not work for software development.

1. **Software Project Management**

**3.1 Objective:**

The Objective of this project was to set goals and achieve those goals in the given time period.

We were in continuous communication with the client and were getting constant constructive feedback.

As a part of continuous development we used to Develop - > Test -> Debug -> Deploy and repeat the same process.

We did proper planning and followed that plan throughout the project**.**

**3.2 Scope**

The scope of project was to develop and deploy an Admission system for the University, about which the details are mentioned in Introduction part.

The purpose of the registration process is to determine which students will be taking courses within the University, and for the administration to keep its records up-to-date. From the student’s point of view, registration enables them to acquire the necessary authorized membership of the University, It is usually the case that students will register for particular courses, or modules, at the same time, and the information collected is used by members of the teaching staff to construct class lists.

For the ease we divided the project in smaller parts i.e. **Design** – **Develop** (includes **Debug**) – **Test** – **Deploy – Maintain.**

**3.3 Cost Estimation**

Cost Estimation is considered as the most crucial of all, as it depends more than one element.

We ensure our end product which is ‘deliverable’ is in reasonable size, so that it does not faces performance issues.

The said software has passed all possible manual testing, it is hack proof as it is using Object Oriented methodology and is using latest hack proof methods such as PDO and MySQLi.

This software is fully open-source, i.e. anyone can use it/ distribute/ and contribute.

The said software has MIT License.

Any additional software used to develop this project had not expense as we have used all open source libraries and software.

Miscellaneous expenses such as Travel, Skill, and Training and support etc. are not part of this project.

We are using CoCoMo (Constructive Cost Model) as a Cost Estimation model.

We estimated out project’s number of lines in thousands i.e. KLOC (Kilo Line Of Code).

1. **Software Risk Management**

In this section we will discuss possible risks which occur or may occur or have occurred during the development of this software. By definition, software risk means, a possibility of suffering from loss in software development process is called a software risk.

Risk always involves two characteristics

1. Uncertainty – the risk may or may not happen; there are no 100% probable risks.
2. Loss – if the risk becomes a reality, unwanted consequences or losses will occurs

It is important to quantify the level of uncertainty and the degree of loss associated with each risk. To accomplish this, different categories of risks are considered

1. Project Risks
2. Technical Risks
3. Business Issues

There is another categorization proposed by Charette.

1. Known risks
2. Predictable risks
3. Unknown risks
4. Unpredictable risks

Generally risk is handled by the Project Manager and the Team, client or end-user has nothing to do with Risk which occurs during development. The known risks which have occurred are as below:

**4.0.1 Compatibility Issues:**

Software which is developed in certain said environment i.e. Windows, Linux or Mac OS is limited to that environment only. We cannot run that software on other platforms. This risk can be managed by making that as platform independent.

**4.0.2 Time Constraint:**

Time is a critical factor in a project, which is limited during the project. We need to deliver the product in the given time. To solve this constraint we divided the work, so that each and every one of us gets equal work and no one gets burdened with work.

**4.0.3 Budget:**

Budget gets decided during the planning phase, and cannot get altered or changed during the project. We cannot get over-budget in the middle of the project. To avoid this issue we did project planning properly and made a project charter.

4**.0.4** **Deployment**

One of the major risks is to deploy the project on the server. Issues on server are bound to happen, issues such as server maintenance, unavailability of the server, human error on the server. This sort of the risk is out of the hand of the developer team or the client, a system administrator is responsible for this.

**4.0.5 Legacy Code**

Once the code is deployed in the production, it is continuous in use. It is developer’s responsibility to maintain the code. The deployed code, which becomes obsolete over the time, is known as legacy code. A developed has to update the code over time, so that it does not becomes obsolete

Above are the “**known**” issues which we faced during the project and resolved for the same.

* 1. **Strategy of Risk:**

In a small project like this, the risks are low and limited but risks still occur. As discussed above the risks which occurred, are of “dual nature”, meaning it affects both the developer (team) and the client.

In order to strategize against risk we follow seven principles of risk management

**4.1.1 Maintain global perspective** – view software risks within the context of system in which it is a component and the business problem that it is a intended to solve.

**4.1.2 Take a forward-looking view** – think about the risks that may arise in the future; establish contingency plans so that future events are manageable.

**4.1.3 Encourage open communication** – if someone states a potential risk, don’t discount it. If a risk is proposed in a informal manner, consider it. Encourage all stake holders and users to suggest risks at any time.

**4.1.4 Integrate** – a consideration of risk must be integrated into the software process.

**4.1.5 Emphasize a continuous process** – the team must be vigilant throughout the software process, modifying identified risks as more information is known and adding new ones as better insight is achieved.

**4.1.6 Develop a shared product vision** – if all stakeholders share the same vision of the software, it is likely that better risk identification and assessment will occur.

**4.1.7 Encourage teamwork** – the talents, skills and knowledge of all stakeholders should be pooled when risk management activities are conducted.

**4.2 Risk Identification**

Risk Identification is a systematic attempt to specify threats to the project plan.

By identifying known and predictable risks, the project manager takes a first step toward avoiding them when possible and controlling them when necessary.

There are two distinct types of risks for each of the categories:

* Generic risks : Generic risks are a potential threat to every software project.
* Product specific : product specific risks can be identified only by those with a clear understanding of the technology, the people and the environment that is specific to the software that is to be built

Above of the identified risks, we identified them mostly while in testing. After identifying the risk or threat, we tackled it for the same in generic manner.

Risk identification is important for a project in order to eliminate risk(s).

One methodology for identifying risks is to create a risk item checklist.

The checklist can be used for risk identification and focuses on some subset of known and predictable risks in the following generic subcategories :

* **Product size** – risks associated with the overall size of the software to be built or modified
* **Business impact** – risk associated with constraints imposed by management or the marketplace
* **Customer characteristics** – risks associated with the sophistication of the customer and the developer’s ability to communicate with the customer in a timely manner.
* **Process definition** – risks associated with the degree to which the software process has been defined and is followed by the development organization.
* **Development environment** – risks associated with the availability and quality of the tools to be used to build the product
* **Technology to be built** – risks associated with the complexity of the system to be built and the newness of the technology that is packaged by the system.
* **Staff size and experience** – risks associated with the overall technical and project experience of the software engineers who will do the work.

**4.2Risk Identification Techniques**

**4.2.1 Documentation Reviews**

The standard practice to identify risks is reviewing project related documents such as lessons learned, articles, organizational process assets, etc.

**4.2.2 Information Gathering Techniques**

The given techniques are similar to the techniques used to collect requirements. Lets look at a few of them:

**4.2.3 Brainstorming**

Brainstorming is done with a group of people who focus on identification of risk for the project.

**4.2.4 Delphi Technique**

A team of experts is consulted anonymously. A list of required information is sent to experts, responses are compiled, and results are sent back to them for further review until a consensus is reached.

**4.2.5 Interviewing**

An interview is conducted with project participants, stakeholders, experts, etc to identify risks.

**4.2.6 Root Cause Analysis**

Root causes are determined for the identified risks. These root causes are further used to identify additional risks.

**4.2.7 Swot Analysis (STRENGTH, Weakness, Opportunities And Threats)**

Strengths and weaknesses are identified for the project and thus, risks are determined.

**4.2.8 Checklist Analysis**

The checklist of risk categories is used to come up with additional risks for the project.

**4.2.9 Assumption Analysis**

Identification of different assumptions of the project and determining their validity, further helps in identifying risks for the project.

**4.2.10 Outputs to Identify Risks**

This process of Risk Identification results in creation of Risk Register.

**4.2.11 Risk Register**

A Risk Register is a living document that is updated regularly throughout the life cycle of the project. It becomes a part of project documents and is included in the historical records that are used for future projects. The risk register includes:

**5. Software Quality Management**

**5.1 Concept**

Software quality management is a management process that aims to develop and manage the quality of software to make sure the product satisfies the user.

The goals of SQM (software quality management) is to make sure the product follows regulations and meets the quality standards expected by the customer.

Software quality managers have to test the product before it is released to the market and they do this by a series of steps called the software cycle in order to reveal and fix bugs before release.

Their job is not only to make sure that their software is in good shape for the consumer, but also to encourage the quality culture to everyone and avoid fraud by protecting their software through proper development.

**5.1.1 Definition**

The aim of Software Quality Management (SQM) is to manage the quality of software and of its development process.

A quality product is one which meets its requirements and satisfies the user.

A quality culture is an organizational environment where quality is viewed as everyone's responsibility.