DELIVERABLE 1
PROJECT PROPOSAL

PART PRO MANAGEMENT SYSTEM

Group Name: Part Pro Management System

Group Members:

S.No	FULL NAME	STUDENT ID	EMAIL
1.	SAI RAHUL PADMA	11714588	sairahulpadma@my.unt.edu
2.	NIKHITA MUVVA	11601586	NikhitaMuvva@my.unt.edu
3.	SNEHA REDDY GANGANNAGARI	11642761	snehareddygangannagari@my.unt.edu
4.	HIMABINDU CHUNDURI	11724307	himabinduchunduri@my.unt.edu
5.	PAVANI VENIGALLA	11697604	PavaniVenigalla@my.unt.edu
6.	SAI SAMYUKTHA PASPULETI	11654305	SaiSamyukthaPaspuleti@my.unt.edu
7.	JHASHA SRI EDE	11685874	JhashaSriEde@my.unt.edu
8.	RISHIKA YALAMANCHILI	11685270	RishikaYalamanchili@my.unt.edu

DESCRIPTION:

A Part Pro Management System website is used to buy vehicle related parts and products. This application will be used by customers and employees. Customers will add their vehicle related details and the relative products will be dynamically displayed in the customized dashboard page. User will checkout the products and complete the payment through payment gateways. Employees will use the application to add products in their inventory by selecting available suppliers and available products from the suppliers. Employees can view and manage the online sales.

Here is the complete description of a part pro website:

1. **Online Product Catalog and Search:**
   Customers can search for products and dynamically products and related categories will be shown in the dashboard. Different types of categories will be available and all the available products in the store will be populated once user lands on dashboard page.

2. **Product Details**:
   Customer can view all the product related details by clicking on each product. He/She can see the availability of the product, product description, customer rating etc.
3. **Store Locator:**
   Customer's current location will be fetched and all the nearest store locations will be shown to the customer. The nearest store location will be displayed on the customer's dashboard page.
4. **Cart Management:**
   Customers can check out the products and can view or manage all the product details in cart page. User can complete the payment once user verifies all the product detail in cart page.
5. **Payment Gateways:**
   User can complete the payment through payment gateway and user can view the confirmation details regarding the payment they have made. If the payment was successful user can view the acknowledgment page, and If payment was declined or pending, warnings will be shown in a UI banner.
6. **Inventory Management:**
   Multiple locations will be available for a store and products will be stalked and multiple types of products will be available in an inventory. Product Quantity and product status like 'Out of Status', 'Available' etc., can be viewed for each product in the specific location.

7. **Employee Management:**
   Different types of roles assignments like Store Owner, Assistant Employee, Associate Employee etc., will be managing the entire store. Store owner will able to manage entire store and will have access to online and offline sales. Associate or Assistant Employees will assist the customers for the orders and payments done by them.

8. **Integration Capabilities:**
   Customers and Employees can sign up for the website and login into the application. Authentication will be done in two steps. Users will login into the application by entering username and password. Once user was able to authenticate successfully, Multi Factor authentication will be done by entering the OTP which was sent to the user's email ID.
   Payment capabilities were also integrated using third party payment gateways.
9. **Customized Shopping:**
   Users will login securely into the application and view all the product details and checkout the required products and complete the payments. The entire process will be simple, smooth and dashboard is customizable as users search for the products and select the required.
10. **User Friendly:**
    The application is responsive and it is user friendly for both customers and employees. The application is cloud native and cloud agnostic. The application can be used in real time as it can be scaled up or scaled down based on the web traffic as the web application is deployed in Azure VM cloud.
11. **Supplier Management:**
    Employees can select all the available suppliers who supplies products to the required locations. Multiple suppliers would have capability of supplying multiple products to each of the locations. Once employee acknowledges the products which needs to be added in the Inventory, inventory will be updated with product quantity and status.
12. **Online Sales Management:**
    Store owner can view and manage all the online sales done by the customers. He/She can view the total price, which product has been sold, which customer has purchased and which employee got engaged to assist the customers.

Technologies Used:

Frontend Stack: HTML, CSS, JavaScript , Bootstrap, React js

HTML: It is used to create and structure the web pages and the applications. It is compatible to all the browsers and all the latest versions.

CSS: It is used to describe the presentation of the document. Material CSS is also used which will be more user friendly, performance of the application will be more optimizable because of well-designed UI components and user experience will be improved more.

JavaScript: It enables client-side script to interact with the user, communicate asynchronously and alter the document content that is displayed. It significantly increases the user experiences and without any need of page reloads.

Bootstrap: It makes the UI more responsive and application will be user friendly. It enables rapid development and significantly reduces the amount of code to build UI interfaces.

React js: it is declarative, efficient, and flexible JavaScript library for building user interfaces. It mainly focuses on developing single page applications with reusable UI components. The main advantage is its virtual DOM that feature optimizes and accelerates the rendering process.

Backend Stack: Core Java, Spring Boot, JPA, MySQL and Docker.

Core Java: It offers robust, secure and platform independent programming language. It enables developers to create cross-platform applications very easily.

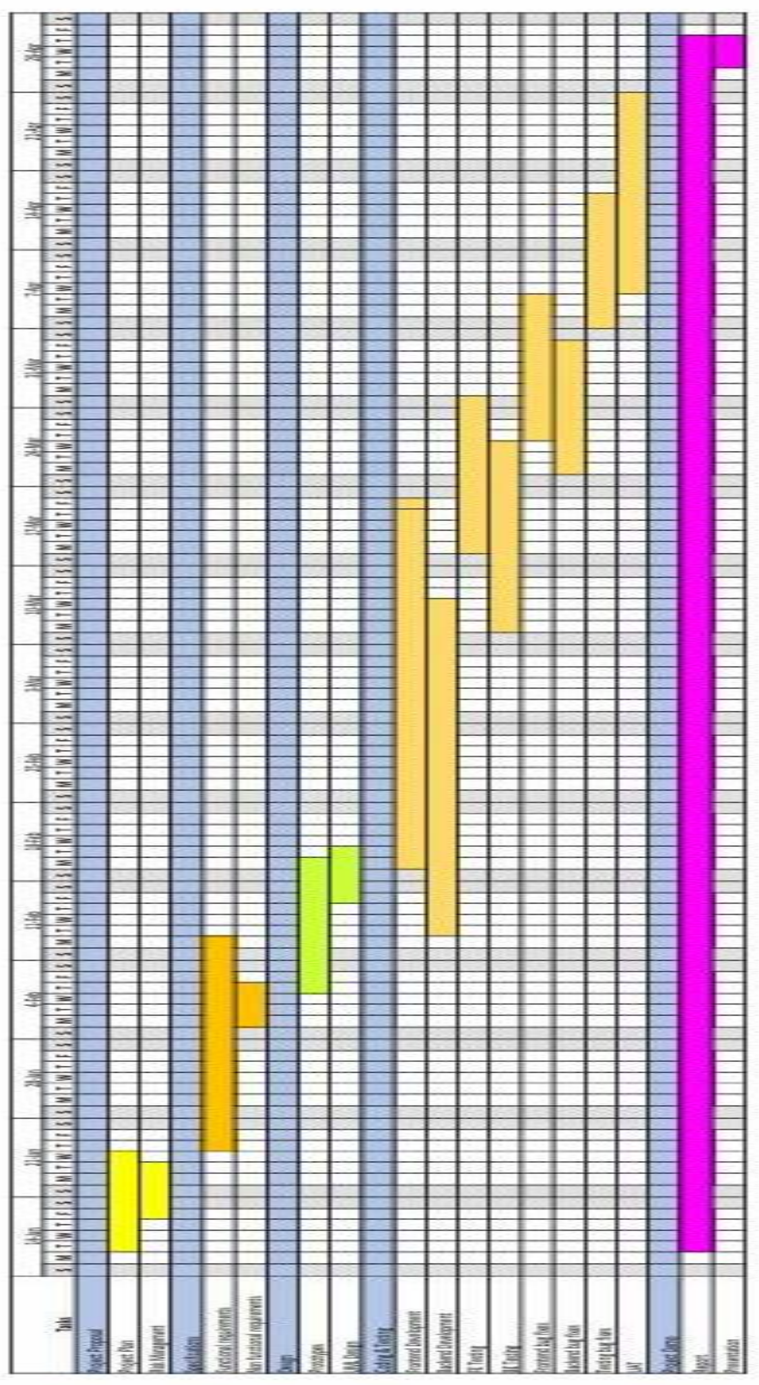
Spring Boot: It is a java spring framework which significantly reduces development time and increases productivity by auto- configuring dependencies, embedded servers for rapid development, testing and deployment.

JPA: It is used for entity relationship management and mapping. It can easily persist, merge update, delete and patch the data into the tables and there by improves productivity because of easy access of the data.

MySQL: It is a relational data base management systen where data is stored in one of the databases, which contains tables and each table contains fields and tuples.

Docker: It is a platform for creating, delivering, and using apps within small, lightweight containers. By separating apps within containers, it improves scalability and efficiency, streamlines configuration management, and guarantees consistency across various development and deployment environments.

GANTT CHART:



This timeline chart is also submitted as separate pdf for more visibility.

RISK MANAGEMENT:

1. RISK IDENTIFICATION:

a. Security Risks:

- **Access Controls:** Multiple role assignments will be done for employees to access data based on roles. Store owner can access any part of data, but the other employees cannot access all the data which helps us from prevention of data loss outside the system. If there are no role assignments, the personal data would be leaked and leads to data breach.
- **Payment Processing Security:** As we are integrating third party payment gateways, if there were any failures in payment processing it would result in economic damage of the users.

b. Performance Risks:

- **Bugs and Errors:** If there are a greater number of bugs which would break main core functionality of the application, it would impact customers user experience as the performance and efficiency of the website decreases.
- **Resource Intensive Operations:** Some of the operations within the application are more resource intensive and more CPU will be utilized which would slow down the system and effects on the performance of the application.
- **Scalability:** If there are a greater number of customers during peak hours, inadequate scalability of the servers would impact customers user experience and it also impacts on the performance.

c. Risks to the User Experience:

- **Poor Usability:** Complex navigations, complex interfaces, and lack of clarity in functionality would impact on usage of website by the users.
- **Cross Browser Compatibility:** Different web browsers can have impact on the user experience. It is essential to have device compatibility.
- **Slow performance and Load Times:** Slow loading of application would also impact on the user experience.

d. Maintenance and Upgrades:

- **Incompatibility Issues:** Version Dependency issues would occur if we moved from one environment to another environment. This would lead to wide range of configuration and user specific bugs.

- **Downtime and Service Disruption:** Unplanned maintenance and upgrades or the extended downtime would impact significantly on users.

e. **Financial Risks:**

- **Fraud and Security Breaches:** Unauthorized access of the application would lead to fraudulent transactions which impacts financial stability and customer trust.
- **Revenue loss Due to System Downtime:** Unplanned outages would significantly lead to revenue loss for the application owners.

2. RISK MONITORING AND REEVALUATION OF RISKS

a. **Security Risks:**

- **Access Controls:**

Risk Monitoring: Conduct regular audits of user roles and permissions to ensure that they are correctly assigned. Setup automated alerts for any unauthorized access or change in role or permission levels.

Reevaluation: Regularly review and update access control policies and permission to adapt if there were new roles.

- **Payment Processing Security:**

Risk Monitoring: Use ML algorithms to detect if there were fraudulent transactions. Do all the third-party payment gateway updates to prevent vulnerabilities.

Reevaluation: Collect feedback from the users regarding the payment processing experiences. Continuously check payment related standards and regulations.

b. **Performance Risks:**

- **Bugs and Errors:**

Risk Monitoring: Implement CI/CD pipelines that include automated testing to identify bugs during development cycle. Use error tracking tools and capture logs in production environments.

Reevaluation: Establish feedback loops with the users to report the bugs. Schedule retrospectives to discuss and mitigate the past bugs and errors.

- **Resource-Intensive Operations:**

Risk Monitoring: Use application performance monitoring tools to understand which resources utilizes more CPU and memory. Regularly perform load testing on the application to understand how application performs in various situations.

Reevaluation: Refactor codes which are inefficient and resource intensive. Reassess and plan for resource allocation and adjust infrastructure based on demand.

- **Scalability:**

Risk Monitoring: Do the stress testing to determine how many users will the application handle during peak hours. Monitor transaction volumes in peak hours to understand trends and metrics.

Reevaluation: Change required architectural improvements to increase scalability of the application. Auto scaling capabilities of the application server should be available to manage based on demand of the customers and web traffic.

c. **Risks to the User Experience:**

- **Poor Usability:**

Risk Monitoring: Regularly perform user acceptance testing to gather feedback on the usability of the website. Use analytics tools to track the user behavior on the website.

Reevaluation: Review the website's accessibility to ensure it meets all the website's standards. Perform the iterative design approach that helps to add customer's feedback in each iterative cycle.

- **Cross Browser Capability:**

Risk Monitoring: Do browser testing in multiple devices and browsers to identify any compatibility issues. Add client-side logging and capture those logs while testing multiple browsers.

Reevaluation: Ensure that application is responsive and it should provide optimal experience in wide range of devices. Always use the latest technology stack which is compatible to all the browser versions.

- **Slow Performance and Load Times:**

Risk Monitoring: Monitor website performance metrics using various software's to identify slow loading pages or elements.

Reevaluation: Optimize images, minify CSS and JS files and implement lazy loading to improve the load times. Regularly review and upgrade hosting solutions and cloud services to increase scalability and speed.

d. **Maintenance and Upgrades:**

- **Incompatibility Issues:**

Risk Monitoring: Implement automated testing in different environments that identify and test different version dependencies and we can identify bugs easily. Use CI/CD pipelines to ensure that changes can be automatically integrated and tested.

Reevaluation: Regularly review and update the version dependencies and configurations or use automated deployment containers like docker which is useful from build till deployment which would take care of version controlling and dependency management in multiple environments.

- **Downtime and Service Disruption:**

Risk Monitoring: Continuous monitoring and alerting systems would easily let us know if there is any service disruption or unplanned outages. Implement and have some backup systems which would replace if there were any failure in main systems.

Reevaluation: List down the insights from the previous unplanned outages and service disruption and use those analysis and insights for future. Notify the users prior for scheduled outages and explain the users which part of the system is getting impacted.

e. **Financial Risks:**

- **Fraud and Security Breaches:**

Risk Monitoring: Implement real time security monitoring to detect for any unauthorized access. Conduct regular security audits to detect for frauds or for any unauthorized access to the application.

Reevaluation: Regularly review and update the security policies and update the protocols whenever there is change in architecture and design.

- **<u>Revenue Loss Due to System Downtime</u>:**

Risk Monitoring: Do system monitoring and keep alerts for entire system and it will notify if it exceeds expected value of thresholds. Write root cause analysis for all the previous issues and bugs which would help us for the future issues.

Reevaluation: Regularly review and update the entire architecture to avoid any security issues and risk of unplanned outages.

3. CONTINGENCY PLANS FOR RISK PREVENTION

a. CONTINGENCY PLANS FOR SECURITY RISKS:

- Create an incident response plan and escalate the incident to the external stakeholders and users.
- Do regular data backup and perform encryption for the data to ensure for the data integrity.
- Create a disaster recover strategy to ensure less unplanned outages.
- Prepare a legal compliance and communication framework and implement a structured approach which would explain for affecting parties.

b. CONTINGENCY PLANS FOR PERFORMANCE RISKS:

- Implement performance monitoring and alerting systems to detect performance gets degraded prior.
- Implement load balancing and resource scaling strategies for better control of web traffic during peak hours.
- Optimize the data base queries and codes and implement caching strategies for better performance of the system,.
- Implement the failure over mechanism for all the service calls and for third party integration services to retry for a specific period and throw the error if it fails instead of more latency and loading issues

c. <u>CONTINGENCY PLANS FOR RISKS TO THE USER EXPERIENCE:</u>

- Create user feedback loops and identify concerns from the users and implement these issues in subsequent development cycles.
- Perform cross browser compatibility testing in all the different browsers and devices and raise the platform bugs to the platform team.
- Optimize the code by implementing lazy loading for populating the data on the page on reaching end of the segment, minify the CSS and JS code and optimize the images.
- Continuously implement the accessibility and usability enhancements for the application for better latest user experience standards.

d. <u>CONTINGENCY PLANS FOR MAINTENANCE AND UPGRADES:</u>

- Plan for scheduled outages for upgrades and updates of the application during off peak hours.
- Use Staging environment or pre-production environment for testing before deploying changes directly to the production environment.
- Quickly roll back the features or bugs from the production environment if they tend to break other features or functionalities after Go live of the application,
- Notify all the users through mails, SMS and push notifications regarding the scheduled planned outages for getting their support.

e. <u>CONTINGENCY PLANS FOR FINANCIAL RISKS:</u>

- Maintain the financial reserve for the system if there is any impact on financial downturns.
- Regularly review the cost management of the system and utilize all the cloud services and implement auto shut down strategies for better cost management.
- Invest some funds on insurance and implement risk transfer strategies if there is any cyber threats and data breaches.
- Create revenue from the application through multiple streams like giving premium features for the customers, pay as you go strategies and subscription models for not impacting financially on the entire systems.

- **ROLES ASSIGNMENT :**

TASK	ASSIGNED TEAM MEMBERS
PROJECT MANAGEMENT LEAD	NIKHITA MUVVA
REQUIREMENTS LEAD	JHASHA SRI EDE
DESIGN LEAD	SAI SAMYUKTHA PASPULETI
IMPLEMENTATION LEAD FOR FRONT END	NIKITHA MUVVA
IMPLEMENTATION LEAD FOR BACK END	SAI RAHUL PADMA
CONFIGURATION MANAGEMENT LEAD	SNEHA REDDY GANGANNAGARI
TESTING LEAD	HIMABINDU CHUNDURI
DOCUMENTATION LEAD	SAI RAHUL PADMA
DEMO & PRESENTATION LEAD	PAVANI VENIGALLA
SYSTEM ADMINISTRATOR LEAD	RISHIKA YALAMANCHILI

LANGUAGE/TECHNOLOGY	ASSIGNED TO
Spring Boot	SAI RAHUL PADMA
CSS	JHASHA SRI EDE
JavaScript, Bootstrap	NIKHITA MUVVA
React JS	RISHIKA YALAMANCHILI
Azure Cloud	SAI SAMYUKTHA PASPULETI
Docker	PAVANI VENIGALLA
Core Java	HIMABINDU CHUNDURI
MySQL	SNEHA REDDY GANGANNAGARI

INDIVIDUAL CONTRIBUTION:

FULL NAME	CONTRIBUTION
SAI RAHUL PADMA	I have designed the architecture of this application and the description and distributed tasks to the team members according to their skills.
NIKHITA MUVVA	I have prepared the Gantt Timeline chart and involved in the presentation.
SAI SAMYUKTHA PASPULETI	I have involved in identifying technologies which would require for this application and helped Nikhita for preparing Gantt charts.
RISHIKA YALAMANCHILI	I worked on reevaluation of the risks that are identified and helped in project presentation.
PAVANI VENIGALLA	I worked on finding the risks involved in the project and helped in project presentation.
HIMABINDU CHUNDURI	I worked on finding contingency plans for risk prevention and helped Pavani for finding risks.
SNEHA REDDY GANGANNAGARI	I worked on finding contingency plans for risk prevention and helped in project documentation.
JHASHA SRI EDE	I worked on reevaluation of the risks that are identified and helped in project documentation.