CMPS310 Software Engineering - Group Project –

Milestone-2:

2) Identify Constraints and categorize them into different groups. Also, recognize the quality requirements/Non-Functional Requirements (NFRs) of the system

Identified Constraints

The constraints are categorized into different groups based on their impact on the design and implementation of the iQVR system.

Constraint	Architecture Requirement
Business Constraints	The system must be integrated with external entities, such as qPay for payments, vehicle manufacturers, and Qatar Trade Service, to ensure proper verification and payment processing.
Development Constraints	The iQVR system must use Java and C for its development, as these are the languages the technical staff are proficient in.
Schedule Constraints	The first version must be delivered within three months, with a full system launch within 12 months to avoid financial penalties.
Technical Constraints	Core vehicle and ownership data must be protected from unauthorized access and kept separate from user interface components to ensure data security.
Resource Constraints	The system is limited to a maximum of 10 additional technical staff and only 20 new servers to manage operations.
Practical Constraints	Only one system administrator will manage the entire iQVR online platform. Budget constraints limit the number of new hires and hardware resources.

Constraints Identified and Categorized

Constraint	Description	Ways to Address Constraint	Technique to Test Constraint
Business Constraints	The system must integrate seamlessly with external entities such as qPay for payment processing, vehicle manufacturers for VIN verification, and Qatar Trade Service.	Establish secure APIs and communication protocols for data exchange. Conduct comprehensive API documentation to ensure proper use.	Perform integration tests to validate data flow between iQVR and external systems, ensuring data is exchanged securely and accurately.
Development Constraints	The system must be developed using Java and C, as the existing technical staff are experienced only in these languages.	Leverage Java and C frameworks and best practices to streamline development while utilizing existing staff expertise.	Code reviews and unit testing will ensure that the system functions correctly, using proper language-specific standards.
Schedule Constraints	The project must deliver the first version within three months, and the entire system must be operational within 12 months. Any delays will incur additional costs.	Apply Agile methodologies, dividing the project into manageable sprints to ensure timely progress and consistent monitoring.	Use sprint retrospectives to evaluate progress, and adjust tasks as needed to ensure the timeline is met.
Technical Constraints	Core vehicle and ownership data must remain secure and be isolated from general user access to prevent unauthorized breaches.	Implement robust data encryption (e.g., AES-256) and role-based access control (RBAC) to safeguard sensitive information.	Perform penetration testing and data security audits to ensure data protection measures are effective.
Resource Constraints	The department is limited to hiring only 10 new technical staff and can allocate a maximum of 20 additional servers.	Optimize the system architecture using load balancing and efficient resource management. Implement cloud solutions if needed to ensure scalability.	Conduct performance and stress tests to verify that the system can handle operational demands within resource limits.
Practical Constraints	The system must be managed by a single system administrator, and financial limitations restrict staff expansion and hardware upgrades.	Automate administrative tasks (e.g., system monitoring, backup processes) to reduce the load on the administrator. Use efficient, self-healing mechanisms.	Test administrative workflows to ensure they can be managed by one person. Simulate failure scenarios to ensure automation works as expected.

Quality Requirements / Non-Functional Requirements (NFRs)

Quality Attribute	Description	Ways to Address	Testing Method
Usability	The interface must be intuitive and require minimal training for users, such as vehicle owners and insurance companies.	Design a simple user interface with clear navigation and concise instructions. Use minimal steps for key tasks.	Conduct usability testing with real users, collect feedback, and make adjustments to improve user experience.
Performance	The system should process key requests, such as registration renewals and payments, within 5 seconds for 90% of cases.	Optimize database queries and use caching mechanisms to speed up response times. Implement efficient algorithms.	Perform load testing to measure transaction times under peak load conditions and identify performance bottlenecks.
Scalability	The system must handle up to 30 million vehicles over the next 10 years, scaling from the current 10 million.	Use a distributed architecture and load balancing to handle increased traffic and data volume.	Simulate high user load and monitor system performance, ensuring it can scale efficiently without degradation.
Modifiability	The system must be easy to update or modify without affecting existing functionality.	Implement a modular design with well-defined interfaces, allowing isolated updates to components.	Perform change impact analysis and verify that modifications do not negatively impact other system components.
Availability	The system must be available 24/7 with an uptime of 99.9%, and downtime should not exceed 2 hours per week.	Use redundancy, failover mechanisms, and eliminate single points of failure.	Conduct failover and recovery tests to ensure the system can handle failures and maintain high availability.
Quality Properties	Attributes like speed, reliability, and robustness, measured using specific metrics (e.g., transaction speed, uptime).	Implement quality control practices and monitor key metrics regularly.	Use performance monitoring tools to measure metrics like speed and reliability. Conduct stress and robustness testing.

Additional Proposed Constraints

Constraint	Description	Ways to Address Constraint	Technique to Test Constraint
Legal Constraints	The system must comply with Qatari data protection laws and regulations to ensure user privacy and data security.	Implement data encryption, privacy policies, and user consent mechanisms to meet legal requirements.	Conduct compliance audits and legal reviews to verify adherence to local regulations.
Hardware Constraints	The system must be compatible with the existing hardware infrastructure, including Oracle-based servers.	Use hardware-optimized algorithms and ensure compatibility testing with existing infrastructure.	Perform hardware compatibility testing to verify that the system operates efficiently on the current servers.
Network Constraints	The system must function efficiently under varying network conditions, especially in areas with limited connectivity.	Optimize data transfer processes and use data compression techniques to handle network issues.	Simulate different network scenarios to ensure the system maintains acceptable performance levels.

Enhanced Quality Requirements / Non-Functional Requirements (NFRs)

1. **Scalability**
   - **Requirement**: The system must support up to 30 million registered vehicles within the next 10 years, handling 10,000 concurrent registration requests during peak periods.
   - **Scenario**: During a registration renewal campaign, the system must process 10,000 simultaneous requests without degradation in performance.
   - **Testing**: Perform load testing using simulated user traffic to ensure that the system maintains efficiency under peak load.
2. **Performance**
   - **Requirement**: Key transactions, such as vehicle registration and payment processing, should be completed within 5 seconds for 90% of requests.
   - **Scenario**: When a user submits a vehicle registration, the system must fetch all necessary records and complete the process in less than 5 seconds.

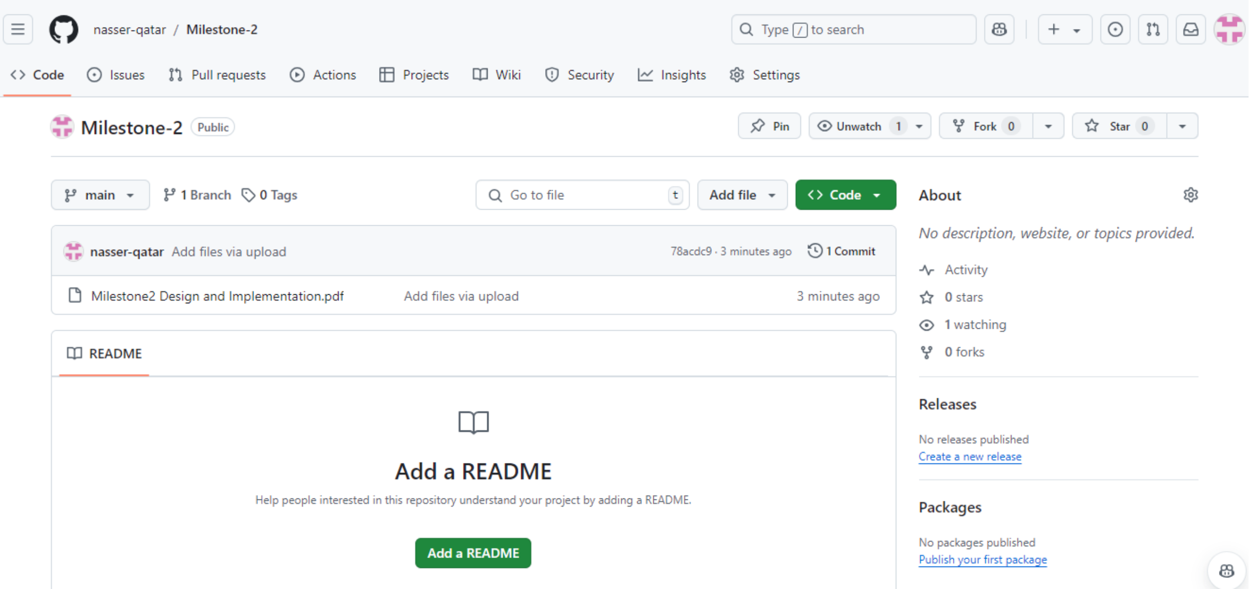
  - **Testing**: Conduct performance benchmarking under various loads to ensure the system meets this criterion.

3. **Availability**
   - **Requirement**: The system must achieve 99.9% uptime, with no more than 2 hours of scheduled downtime per week.
   - **Scenario**: During a critical maintenance window, the system must continue functioning using failover servers without disrupting service.
   - **Testing**: Simulate server failures and measure the system's recovery time to confirm high availability.
4. **Security**
   - **Requirement**: All sensitive data, including vehicle and owner information, must be encrypted using AES-256 encryption and protected by role-based access controls.
   - **Scenario**: If an unauthorized access attempt is detected, the system must log the incident and block access immediately.
   - **Testing**: Conduct penetration testing and review system logs to ensure security measures are effective.
5. **Data Integrity**
   - **Requirement**: The system must ensure that all transactions are recorded accurately, with automatic rollback mechanisms in case of failure.
   - **Scenario**: If a payment fails during the registration process, the system should revert to the previous state without data loss.
   - **Testing**: Simulate transaction failures and verify that the system maintains data integrity.
6. **Portability**
   - **Requirement**: The system must be accessible from various devices, including desktops, tablets, and smartphones, with a consistent user experience.
   - **Scenario**: Users should be able to complete registration renewals using any device without a change in functionality.
   - **Testing**: Conduct cross-platform testing on different devices to ensure compatibility and performance.
7. **Usability**
   - **Requirement**: The user interface must be simple and intuitive, allowing users to complete tasks with minimal training and support.
   - **Scenario**: A vehicle owner should be able to renew their registration in three clicks or fewer, with clear guidance at each step.

- **Testing**: Conduct usability tests with a diverse group of users and gather feedback to refine the interface.

8. **Modifiability**

- **Requirement**: Future system updates or new features must be implementable within 4 weeks without impacting existing functionality.
- **Scenario**: If a new regulation requires changes to the registration process, the system should be updated and deployed within the specified timeframe.
- **Testing**: Perform change impact analysis and regression testing to ensure modifications are smooth and effective.



nasser-qatar Add files via upload 78acdc9 · 3 minutes ago 1 Commit

Milestone2 Design and Implementation.pdf Add files via upload 3 minutes ago

No description, website, or topics provided.

No releases published
Create a new release

No packages published
Publish your first package

Add a README

Help people interested in this repository understand your project by adding a README.

https://github.com/nasser-qatar/Milestone-2.git