Explore MQ

Presentation

Sathya Reesha Kodali
Sathvika Mannepally
Mariya Hoque Trina
Lisa Thirapala

Introduction

Explore MQ

- Interesting & dynamic experience
- Encourages community spirit
- Thrill of campus exploration with ease
- Adaptable challenges
- Encourages collaboration
- Alluring incentives (prizes etc)
- Enjoyable, competitive & instructive

System Design Document

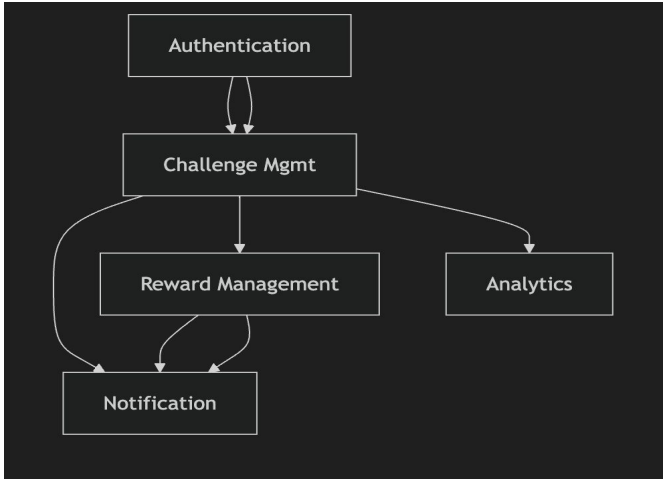
1. System Architecture:Mobile app (iOS/Android) for user interaction, with a backend server that manages authentication and challenges and integrates with the university's SSO.

2. Data Management:User and challenge data are stored in a relational database (MySQL/PostgreSQL), encrypted with AES-256 and has regular backups for recovery.

3. API Gateway and Integration:API endpoints provide secure data management and integration with external systems for authentication and award tracking.

4. Performance Trade-offs:We are balancing immediate notification and system reactivity against the complexities of incorporating external reward systems.

5. Subsystem Overview:Authentication, Challenge Management, Reward Management, Analytics, and Notifications are essential for a flawless user experience.



Data Definitions

1. User Identification: Each user is assigned a unique UserID, UserName, and Email address for login and communication.

2. Profile and Role: ProfileType categorises people as students, employees, or store partners, while Role specifies access levels.

3. Task Tracking: Each task is assigned a unique ChallengeID and ChallengeName, with TaskProgress indicating completion status.

4. Rewards & Security: After completing a job, users receive a RewardVoucher, an authToken for secure access, and LocationData in JSON format for visited locations.

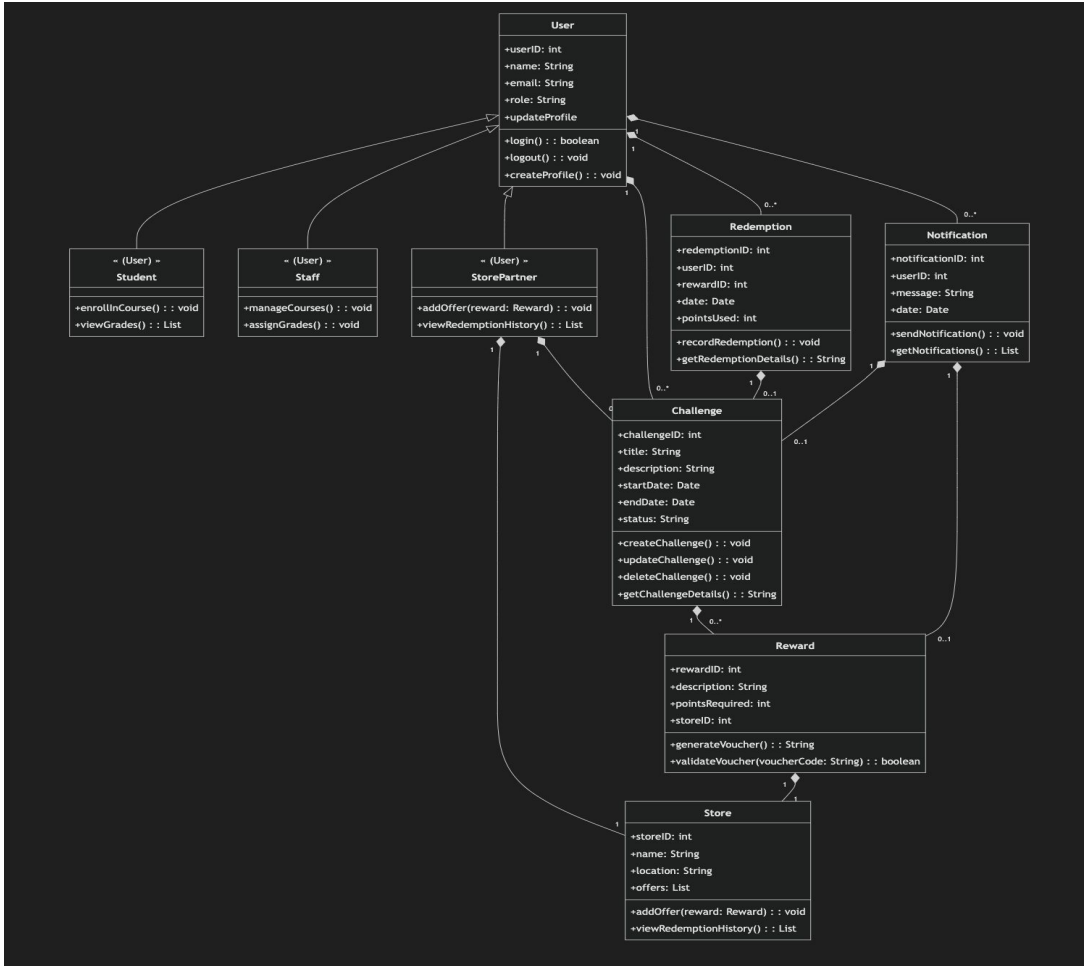
Field Name	Type	Meaning	Example
UserID	INT	Each student or staff or store partners have been assigned a unique identifier.	123456
UserName	VARCHAR(50)	The user's MQ login credentials.	s124345
Email	VARCHAR(50)	The user's MQ email.	@mq.edu.au
ProfileType	ENUM('student', 'staff', 'store partner'	Determines if the user is a student or staff member or store partner.	staff
ChallengeID	INT	A unique identifier for each challenge.	2345342
ChallengeName	VARCHAR(50)	Name of the challenge in which the student is participating.	Locating the campus facility.
TaskProgress	ENUM('complete', 'incomplete')	Tracks the current status of the challenges done by students.	incomplete
RewardVoucher	VARCHAR(100)	A voucher code is generated when a student completes a task.	Sq123
StoreID	INT	Unique identifier for store partner.	4783792
ChallengeDate	DATE	The date challange was issued or expired.	2024-10-24
authToken	VARCHAR(256)	Token used to authorise the users for securing the information.	f237js342edj34
LocationData	JSON	Storing the data on the locations that was visited during a challenege.	{"Mason Theatre"}
Role	ENUM('student', 'admin', 'store Partner')	Defines the level of user access.	student

Class Diagram

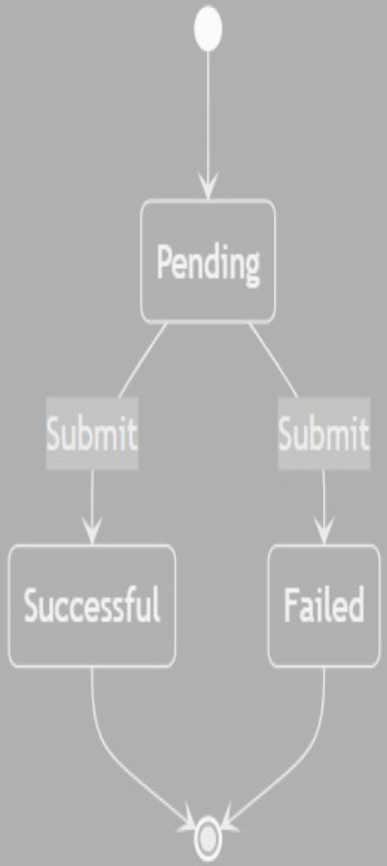
Emphasise the following main classes: Challenge, Reward, Store, Redemption, Notifications, Staff, Student, User, and StorePartner.

Bring up inheritance: StorePartner, Staff, and Students all inherit from the user.

Associations and Multiplicities



State Diagram



1. Challenge states: Draft, Active, Finalised, and Expired
2. Redemption states: Pending, Successful, Fail
3. Rewards states: Available, Redeemed, Expired

Requirements Traceability Matrix

1. Requirement Identification: Each need is assigned a unique ID (e.g., FR-1, FR-2) corresponding to specific functionality like Login, Challenge Enrolment, and Task Tracking.

2. Use Cases and Classes: Each need describes relevant use cases (e.g., Login, Notifications) and associated classes (e.g., User, Challenge) to help illustrate the system's interactions and structure.

3. Methods and Packages: Methods (such as login() and enrollChallenge()) describe the individual operations implemented within the code and are organised under relevant packages (such as Authentication and Challenge Management).

4. Build Number: This column tracks the build version associated with each need, ensuring that it corresponds to the development phases and allows for proper version management.

Requirement ID	Use Cases	Classes	Methods	Packages	Build Number
FR - 1	Login, Forgot Password	User, Auth	login(), resetPassword()	Authentication	
FR - 2	Challenge Enrollment	Challenge, Task	enrollChallenge()	Challenge Management	
FR - 3	Task Progress Tracking	Task, Progress	taskProgress()	Challenge Management	
FR - 4	Reward Redemption	Reward, Voucher	redeemVoucher()	Reward System	
FR - 5	Store Management	Store, Reward	manageStoreRewards()	Reward System	
FR - 6	Notifications, rewards	Notifications, task	sendReminder()	Notification System	
FR - 7	Participation Tracking	Analytics	generateReport()	Analystics	
FR - 8	Student Verification	User, Auth	verifyStudent()	Authentication	
FR - 9	Profile Management	User, Task	manageProfiles()	User Management	

Test specifications:

1. Provide test case specifications together with inputs, outputs, descriptions, and IDs.
2. Emphasise the following elements of the test plan: schedule, resources, milestones, and deliverables.

Minimal Viable Product(MVP):

1. Important features include: Redemption tracking, reward management, challenge enrolment, profile administration, user authentication, and basic alerts
2. Crucial architecture: Challenge and Reward, User Management System database, notification system, subsystems, and basic frontend interface

Milestones and Tasks

Milestones 	Tasks 
1. Profile creation and management	- Prompt box for details
2. Challenge creation and management	- Building challenge creation module
3. Student Participation and Feedback	- System to enroll students for tasks - Asking feedback upon completion of tasks
4. Implementing challenge completion system	- Generating valid barcodes
5. Setting up the UI/UX for user acceptance	- Usability tests - Improving UI/UX as per feedback
6. Testing features	- Testing errors - Deploying initial launch - Collecting initial feedbacks

Risks & Risk Mitigation Strategies

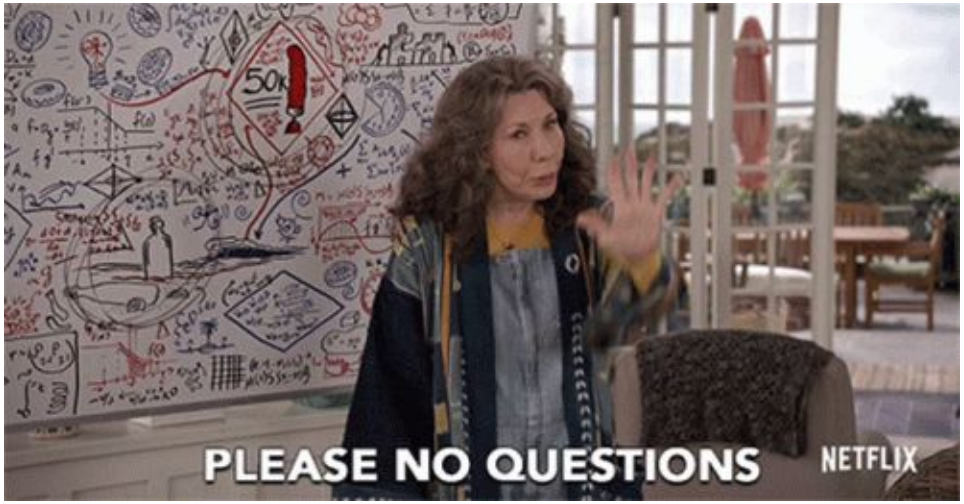
Risk	Risk Management
Change in project goals/stakeholder requirements.	Regular meetings with stakeholders.
Incomplete or misunderstood requirements.	Review requirement analysis.
External systems/teams integration.	Regular integration testing & communication with teams.
Outdated or malfunctioning development tools	Performance testing & scalable technology stacks.
Scope or objective changes mid-project.	Formal process to manage adjustments.
Frequent requirement changes.	Agile methodologies to manage & document changes.
Hardware/software limitations.	Scalable technology stacks & performance testing.
Incompatibility between software tools.	Standardised tool usage among teams.

Conclusion

The main objectives are:

- To create smooth student engagement through dynamic challenges
- To have reliable log in procedures (SSO)
- To provide staff an adequate system to develop and manage challenges
- To prioritize performance, security, and scalability
- To enable constant updates to achieve user acceptance

Do you have any questions?



Thank you!