Title page: Proposal for BIS HACKATHON

Introduction:

This proposal presents an exciting game app development project for the Bureau of Indian Standards (BIS) Hackathon, designed to make learning about Indian standards both fun and engaging. Our mission is to create an interactive educational experience that brings the role of BIS standards to life in a way that resonates with users of all ages.

The game will transform complex standards into fun challenges, real-world scenarios, and exciting quizzes that are easy to understand and relate to. By playing, users will not only learn about the importance of BIS standards but also experience how these regulations ensure safety, quality, and efficiency in everyday products and services.

With a special focus on students, kids, and young professionals, this app will make learning about standards informative, immersive, and entertaining. It's more than just a game—it's an opportunity to cultivate a deeper appreciation for the vital role standards play in shaping our world.

Through this innovative app, we aim to reach a wide audience, fostering a culture of quality and safety in India and encouraging the adoption of BIS standards in a fun, modern, and engaging way.

Title of report: GET STANDARDIZED

Name of software: Unreal game engine

Developer: CODE-ZEE

College Name: Sri Venkateswara College of Engineering, Chennai

---

Walkthrough of the Software Application

1. User Onboarding

Welcome Screen: The app opens with an engaging welcome screen, showcasing the BIS logo and a brief introduction about the importance of Indian Standards.

User Profile Creation: Players are prompted to create a profile by entering their name, sex and age which ensures the player's authenticity and eligibility.

Tutorial: A short, interactive tutorial guides the user through the basic gameplay mechanics while providing an overview of how the game relates to Indian Standards through Flow chart.

2. Main Menu

Game Modes: The user is directed to a central hub that offers multiple game modes:

Story Mode: A narrative-driven journey where players explore different industries and sectors, completing levels by learning and applying Indian Standards.

Challenge Mode: Timed tasks that test the player’s knowledge of various standards, with rewards for high scores are the main attraction of this mode.

Explore Standards: A library mode where users can browse through simplified descriptions of key Indian Standards, with interactive elements to make learning engaging.

3. Gameplay Overview

Level Design: In Story Mode, each level focuses on a specific industry (e.g., electronics, construction, healthcare) and its associated standards. Players must navigate various tasks, such as:

Problem Solving: Players are presented with real-life scenarios (e.g., buying a product, setting up a factory, or building infrastructure) where they need to choose the right standards to ensure safety and quality.

Progression System: Players earn points, badges, and certifications as they progress through the game, with achievements tied to understanding different sets of standards.

4. Interactive Learning and Feedback

Feedback: After completing all the task the app provides detailed feedback on the correct answers, along with explanations about the relevance of the standard involved.

Educational Pop-ups: Throughout the game, users will encounter short, digestible facts about BIS and its role, ensuring that educational content is seamlessly integrated into gameplay.

5. Rewards and Leaderboard

Rewards: Players can earn virtual rewards like badges, certificates, by completing levels and scoring well in tasks. Special rewards like extra lifeline are given for understanding critical standards .

Leaderboard:[Futuristic approach/plans] A global and local leaderboard system encourages users to compete with others in mastering standards, adding a social aspect to the learning experience.

6. User Support and Updates

Help Section: A dedicated help section offers users a guide to the gameplay and to address the inconvenience caused for the user.

7. Final Assessment and Certification

Final Challenge: At the end of the Story Mode, users face a comprehensive final challenge that covers all the standards explored throughout the game.

BIS Certification: Upon successful completion, players may receive a virtual BIS Certificate as *STANDARDIZED CITIZEN* demonstrating their knowledge of Indian Standards, which they can share on social media or download for personal use.

Software Overview

a. Description

"GET STANDARDIZED" will be a 3D adventure game where players assume the role of a character navigating through challenges that require knowledge of Indian Standards. The game will feature rich storytelling, character development, and multiple endings based on player choices.

b. Features

1. Interactive Learning
The game offers a narrative-driven Story Mode where users explore different industries, completing tasks that involve applying BIS standards. Users learn through interactive problem-solving, quizzes, and mini-games.

2. Multiple Game Modes

Story Mode: Focuses on real-world applications of standards across industries.
Challenge Mode: Offers timed quizzes and puzzles to test users' knowledge of Indian Standards.
Explore Mode: A library that provides simplified explanations of key standards and their uses.

3. Progression System
Players can earn points, badges, and virtual certifications as they complete tasks, with levels unlocking progressively. This gamified approach encourages users to learn while enjoying the process.

4. Educational Content Integration
Short pop-ups, questions and feedback systems help users learn about standards while playing, seamlessly blending education with entertainment.

5. Global Leaderboard[Future Plan]
The app features a leaderboard where users can compete with friends and other players worldwide, fostering a competitive learning environment.

6. Rewards and Achievements
Virtual rewards such as certificates, and in-game badges motivate users to continue progressing and mastering Indian Standards.

7. Regular Updates
New levels, industries, and BIS standards are added through periodic updates to keep the content relevant and engaging.

8. User-Friendly Interface
Designed with intuitive navigation and appealing graphics, the app ensures ease of use for a wide range of users, from students to professionals.

c. System Requirements

Mobile Platforms:

Operating Systems: Android

RAM: Minimum 2 GB for smooth gameplay

Storage: At least 500 MB of free space required for initial download and updates (advised)

Desktop Platforms:

Operating Systems: Windows 11

RAM: Minimum 4 GB

Storage: 1 GB of free space

Internet Connection: Required for leaderboard access and content updates.

Development Process

a. Methodology

Simple Single Player Methodology

The single-player game will follow a "Level Progression" methodology.

In this approach, the player starts at the easiest level, where they are introduced to basic concepts about Indian standards. As the player completes each level, they unlock the next, which introduces more advanced topics and challenges.

Each level consists of

1. **Learning Segment:** Short, easy-to-understand information about a specific BIS standard.

2. **Question and tasks:** Players answer questions or solve simple tasks based on what they just learned.

3. **Rewards and Progress:** Players earn points and badges for correct answers and completing levels.

The game keeps track of the player's progress and encourages them to keep playing by gradually increasing the difficulty, making learning fun and motivating them to complete all levels. This method ensures that players learn at their own pace while staying engaged and entertained.

1. Sprint-Based Development
The project will be divided into multiple sprints, each focused on specific features or game levels. Regular reviews will ensure the game remains engaging while meeting educational goals.

2. Testing & Continuous Integration
Regular testing will be carried out at each stage to identify bugs and improve gameplay. Continuous integration will ensure that new features are seamlessly added without disrupting existing functionality.

b. Tools and Technologies

1. Game Development Platform Unreal Engine:

Unreal Engine is selected for its unparalleled graphics capabilities, cutting-edge physics simulations, and seamless cross-platform deployment across Android, iOS, and other platforms. Its powerful rendering engine ensures a stunning visual experience.

Programming Language C++/Blueprints: Unreal Engine's primary scripting languages for building game logic, interactions, and integrations. C++ provides high-performance capabilities, while Blueprints offer a visually intuitive alternative for designers and developers.

2. Design and Animation Tools

CANVA: For designing user interfaces, characters, and game assets.
Blender: For creating any 3D models or animations needed in the game.Figma: Used for UI/UX design and wireframing, ensuring a smooth user experience from start to finish.

3. Backend and Database

Firebase: For user authentication, real-time data, cloud storage, and analytics.
Node.js with Express: For any server-side logic and managing in-game data (leaderboards, rewards, etc.).
MongoDB: A NoSQL database for storing user progress, achievements, and game data, ensuring high performance and scalability.

4. Testing Frameworks

Unreal Engine Testing Framework: For unit and integration testing within the Unreal Engine environment.

Automation Testing (Unreal Engine's built-in testing tool): For automated testing of game logic, physics, and graphics.

Selenium/Appium: For UI testing across different devices and screen sizes.

c. Version Control

To manage the code and collaborate efficiently, Git will be used as the version control system, with GitHub as the repository hosting platform.

1. Branching Strategy

Master Branch: The production-ready code is kept here. Only thoroughly tested and reviewed features are merged into this branch.

Development Branch: Used for integrating features currently under development. This branch will frequently be tested and will serve as a staging area before merging with the master.

Feature Branches: Each feature or improvement will have its own branch, named based on the feature (e.g., "login-system", "leaderboard-feature"). This allows for parallel development of multiple features without conflicts.

2. Code Reviews (Future Plans)

Pull Requests: Before any code is merged into the development or master branches, a pull request will be submitted. Senior developers or team leads will review the code for quality, functionality, and adherence to the coding standards.

Automated Code Analysis: Tools like SonarQube will be used to analyze the codebase for potential bugs, security vulnerabilities, and code smells during the review process.

3. Backup and Version History
Git's version history will allow the team to roll back to any previous version in case of issues or bugs in the current build. Regular backups of the repository will be made to ensure code safety and integrity.

This combination of Agile methodology, robust tools, and version control strategies ensures efficient development, regular updates, and a high-quality final product.

Implementation Details

a. Key Algorithms

1. Level Progression Algorithm

This algorithm ensures that users progress through the game based on their performance in questions, tasks, and mini-games. It adjusts the difficulty dynamically depending on how well users are performing.

Steps:

1. Track user scores for each level.

2. Set thresholds for unlocking the next level based on scores.

3. Adjust the difficulty (time limits, question complexity) based on previous performance.

4. Unlock new levels and tasks when performance criteria are met.

2. Questions Generation Algorithm

Randomly generates quiz questions from a predefined pool based on the user's current progress in the game. It ensures that questions are relevant to the standards and topics learned in earlier stages.

Steps:

1. Based on the user’s current level, filter a pool of questions tagged with relevant standards or topics.

2. Randomly select a subset of questions, ensuring no repetition within a certain timeframe.

3. Assign difficulty levels to questions depending on the user’s performance.

4. Track and store which questions have been answered correctly or incorrectly to adjust future quizzes.

3. Reward System Algorithm

Awards points, badges, and certificates based on user performance and achievements.

Steps:

1. Assign points to each challenge and quiz question based on difficulty.

2. Track cumulative scores for the user.

3. When cumulative scores reach certain thresholds, unlock rewards such as badges or virtual certificates.

4. Update the global leaderboard based on scores.

4. Leaderboard Algorithm[Future Plan]

Ranks users globally and locally based on their scores, encouraging competition.

Steps:

1. Store and track user scores in a database.

2. Compare each user's score against the global score list to determine ranking.

3. Update the leaderboard in real-time as users complete new levels and challenges.

4. Provide filters for users to see rankings based on location, friends, or globally.

5. Content Update Algorithm

Ensures that new standards and updates are pushed to users without interrupting their current gameplay.

Steps:

1. Schedule regular updates for new levels, standards, or quizzes.

2. Notify users when new content is available.

3. Store updates on the server, and download them in the background when the app is in use.

B. Code Structure

Main Components

- Unreal Engine Core: Handles rendering, physics, and core gameplay mechanics.
- GameManager: Centralized control for managing game states, levels, and user progression.
- LevelManager: Responsible for loading and controlling individual levels, their elements, and interactions.
- UIManager: Manages all user interface elements like menus, leaderboards, and settings.
- QuizManager: Handles quiz generation, grading, and progression logic.
- RewardManager: Manages user rewards, badges, and certificates.

Code Modules

- UserModule: Handles user authentication, profile management, and progress tracking.
- GameplayModule: Core game logic for levels, challenges, and interactive tasks.
- QuizModule: Contains logic for generating and grading quizzes and challenges.
- LeaderboardModule: Manages user rankings and real-time updates.
- DataModule: Interfaces with the backend for storing user data, progress, and app updates.

Directory Structure

- /Source
  - /Content -> Game assets like images, sounds, 3D models.
  - /Classes -> All C++ classes for game logic.
  - /Managers -> Core managers for game state control (GameManager, UIManager, etc.).
  - /Gameplay -> Specific gameplay mechanics (levels, challenges).
  - /UI -> Blueprints controlling UI elements.
  - /Data -> Backend interaction and user data management.
  - /Resources -> Non-code resources (quiz questions, images, educational content).
  - /Tests -> Unit and integration tests for the app.

Data Structures

1. User Profile Structure

```
struct UserProfile {
    FString username;
    int32 userID;
    int32 currentLevel;
    int32 totalScore;
    TArray<Achievement> achievements;
    DateTime lastLogin;
};
```

2. Level Structure

```
struct Level {
    int32 levelID;
    FString industryType;
    TArray<Challenge> challenges;
    TArray<QuizQuestion> quizPool;
    bool isLocked;
};
```

3. Quiz Question Structure

```
struct QuizQuestion {
    FString questionText;
    TArray<FString> options;
    int32 correctAnswerIndex;
    int32 difficultyLevel;
};
```

4. Leaderboard Structure

```
struct LeaderboardEntry {
    int32 userID;
    FString username;
    int32 totalScore;
    int32 rank;
};
```

5. Backend Data Structure (e.g., MongoDB)

- User Collection: Stores user profiles, including progress and scores.
- Leaderboard Collection: Stores global leaderboard rankings and user scores.
- Content Collection: Stores game content like quizzes, levels, and educational materials.

Testing and Quality Assurance

a. Test Cases

Test cases are essential to ensure that the game performs correctly, is user-friendly, and delivers an educational experience as intended. Here are key areas where test cases will be applied:

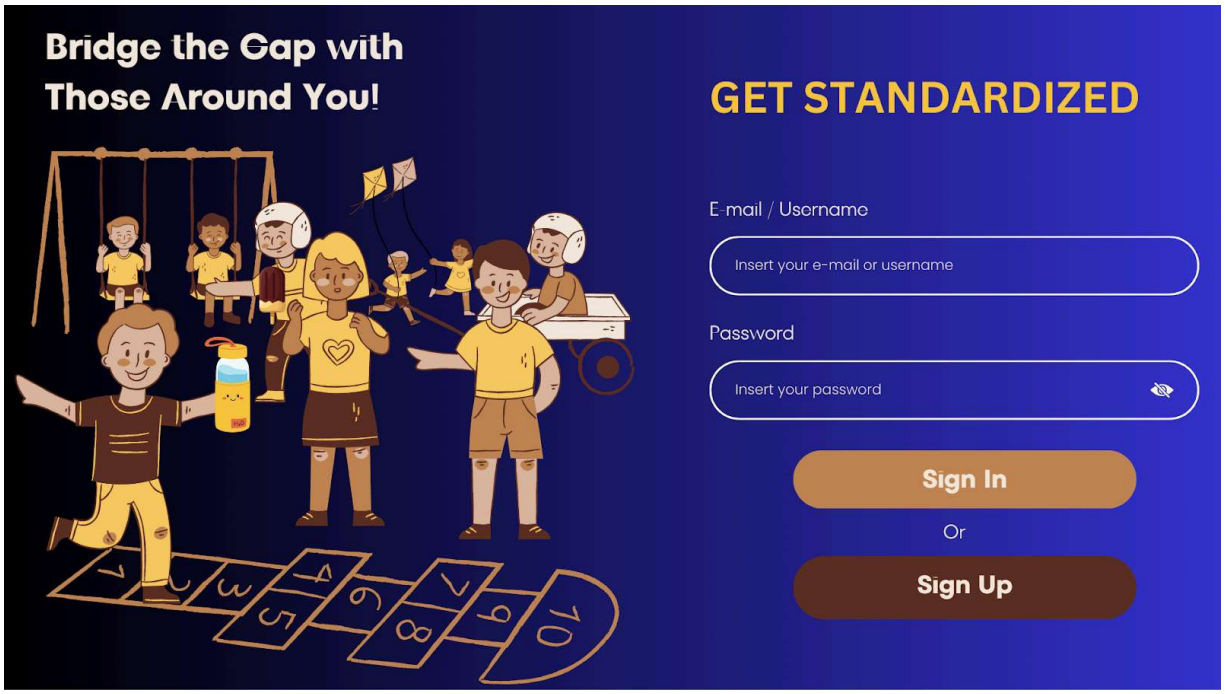
1. Functional Testing

User Registration and Login:

Test Case: Verify that users can create profiles, log in, and log out successfully.

Pros: Ensures smooth onboarding for all users.

Cons: May require constant updates if new authentication methods are added.



Level Progression:

Test Case: Verify that levels unlock correctly based on user performance and scores.

Pros: Guarantees a structured learning experience and user engagement.

Cons: Complex to maintain when scaling new levels or introducing dynamic difficulties.

Quiz System:

Test Case: Ensure quiz questions are generated correctly, and users receive feedback on their answers.

Pros: Confirms accurate educational content delivery.

Cons: If the question pool grows large, ensuring diversity and non-repetition of questions may become challenging.

2. User Interface (UI) Testing

Menu Navigation:

Test Case: Verify that all menus (Main, Settings, Leaderboard, etc.) are accessible and function properly.

Pros: Ensures a smooth and intuitive user experience.

Cons: UI testing across multiple devices and screen sizes can be time-consuming.

Rewards System:

Test Case: Verify that points, badges, and certificates are awarded correctly after completing challenges.

Pros: Helps maintain user motivation and engagement.

Cons: Can lead to bugs if not synchronized well with back-end data.

3.**Educational Accuracy Testing**

Content Review:

Test Case: Verify that all educational material on BIS standards is accurate and up-to-date.

Pros: Ensures the app fulfills its primary educational objective.

Cons: Requires frequent updates as new standards are introduced by BIS.

b. Bug Tracking

To ensure a smooth and error-free experience, a robust bug-tracking system will be implemented.

1. Tools:

Jira or Trello: These tools will be used to log, track, and manage bugs throughout development.

Pros: Provides detailed reporting, prioritization, and tracking of bugs, ensuring they are addressed promptly.

Cons: Can become overwhelming if too many issues are reported at once, requiring strict prioritization.

2. Bug Reporting:

Bugs will be categorized by severity (e.g., critical, high, medium, low).

Critical Bugs: Prevent gameplay or core functionality (e.g., login failure).

High Bugs: Affect game mechanics or user experience (e.g., incorrect scoring).

Medium Bugs: Minor issues that don't impact gameplay significantly (e.g., UI glitches).

Low Bugs: Cosmetic or non-essential issues (e.g., typos).

Pros: Categorization helps the team focus on the most important issues.

Cons: Determining bug severity can sometimes be subjective, leading to misprioritization.

3. Bug Life Cycle:

Bugs will go through the following stages: Reported → Assigned → In Progress → Fixed → Verified → Closed.

Pros: Ensures all bugs are accounted for and resolved in a systematic way.

Cons: Lengthy bug life cycles can delay feature releases if bugs are complex or difficult to reproduce.

c. Performance Metrics

Measuring the app's performance is crucial to ensure that it runs smoothly across all platforms and devices.

1. Frame Rate (FPS)

Metric: The app should maintain a minimum of 30 FPS, with an optimal target of 60 FPS for smooth gameplay.

Advantages: Ensures a responsive and fluid user experience, especially during interactive tasks and quizzes.

Disadvantages: Devices with lower processing power may struggle to maintain these rates, requiring optimization and compromising visual quality.

2. Load Time

Metric: The app’s load time should be under 5 seconds for major screens (e.g., login, level loading).

Advantages: Short load times improve user retention and reduce frustration.

Disadvantages: As content grows (e.g., adding new levels), optimizing load times can become increasingly difficult.

3. Memory Usage

Metric: Memory usage should be optimized to prevent crashes, especially on devices with limited RAM.

Advantages: Optimizing memory usage ensures compatibility with a wide range of devices.

Disadvantages: Balancing rich content (graphics, animations) with low memory consumption can be challenging, requiring trade-offs in performance.

4. Battery Consumption

Metric: The app should not consume more than 10-15% battery during 30 minutes of continuous gameplay.

Advantages: Low battery consumption makes the app more usable for extended periods.

Disadvantages: Battery optimization can reduce performance or limit certain features (like background music or animations).

5. Error Rate

Metric: Track the percentage of crashes, freezes, or errors during gameplay. Ideally, this should be below 1% per session.

Advantages: A low error rate ensures that the app is stable and reliable, leading to better user reviews.

Disadvantages: As more features are added, maintaining a low error rate can become harder, especially with frequent updates.

Conclusion

A comprehensive approach to testing and quality assurance is crucial to the success of the BIS awareness game app. While there are clear benefits to ensuring thorough testing, bug tracking, and performance optimization, these processes require careful planning and can become time-consuming. However, by addressing these areas early and consistently, the final product will be stable, engaging, and educational.

Deployment

a. Deployment Process

1. Build Preparation:

Finalizing Code: Ensure all features, fixes, and tests are integrated into the master branch of the version control system (GitHub/GitLab).

Build Generation: Use Unity's Build Settings to generate platform-specific builds (Android, iOS) for deployment. Each platform may require specific configurations, such as Android manifests or iOS provisioning profiles.

2. Continuous Integration (CI):

Set up CI pipelines (using Jenkins, GitLab CI/CD) to automatically build the game after every code push to the master branch. This ensures that any last-minute changes are built and tested automatically.

CI tasks include running automated tests, checking for code quality (using SonarQube), and generating platform-specific builds.

3. App Store Deployment (Android & iOS):

Google Play Store (Android):

1. Create a Google Developer account.

2. Upload the APK file to the Google Play Console.

3. Complete the listing, including game descriptions, images, and age ratings.

4. Submit for review and approval.

5. Full Release:

Once the app passes the review process and beta testing feedback is incorporated, push the app for a public release on both platforms.

Post-launch updates will be scheduled to introduce new levels, standards, or bug fixes as needed.

b. Environment Setup

To ensure seamless deployment, various environments will be set up and configured for different stages of development and testing.

1. Development Environment:

Tools:

- Unreal Engine: Primary development tool for building the game.
    - - Git: Version control to collaborate on code.
- Visual Studio Code or Visual Studio: Integrated Development Environment (IDE) for coding.
- Local Database: For local testing, such as SQLite or MongoDB.

Features:

- Fast iteration and testing.
- Real-time debugging.
- Local tests on Windows, macOS, Android, and iOS platforms.

1. Staging Environment:

Purpose: A clone of the production environment, where final integration and testing take place before the full release.

Configuration:

- Use cloud databases like AWS or Google Cloud for real-time testing.
- Connect the staging app to live servers.
- Deploy builds to Epic Games Store, Google Play Beta (Android), and TestFlight (iOS) for testing.

Advantages:

- Identifies any issues not caught in development (e.g., network issues, server responses).
- Allows real-world scenario testing with real data and users.

Additional Environments:

1. Testing Environment:

- Utilize Unreal Engine's built-in testing tools, such as Automation Testing.
- Conduct unit testing, integration testing, and UI testing.

1. Production Environment:

- Configure for scalability and performance.
- Monitor performance metrics and analytics.

Unreal Engine Specific Features:

- Utilize Unreal Engine's Multi-User Editing for collaborative development.
- Leverage Unreal Engine's Deployment features for streamlined deployment.
- Use Unreal Engine's Performance Monitoring tools for optimization.

3. Production Environment:

Configuration:

Firebase (Backend): Set up for handling live users, storing progress, handling leaderboards, and syncing data.

App Store / Play Store: Finalized configuration and submission of builds.

Post-launch, continuous monitoring of the production environment will ensure performance and bug fixes are managed efficiently.

4. Backend Services Setup:

Firebase Authentication: To manage user logins, profile data, and cloud saving.

Firebase Realtime Database/Firestore: For real-time syncing of user progress, leaderboards, and challenges.

Firebase Cloud Messaging: To send push notifications about new updates, levels, or achievements.

This environment will be replicated across development, staging, and production with different configurations for each environment to prevent data overlap.

c. Post-Deployment Checklist

1. Functional Validation:

Ensure that all features, including registration, login, level progression, rewards, and leaderboards, are functioning as intended in the live environment.

Perform a final round of testing on actual devices to verify UI responsiveness, gameplay, and educational content.

2. Performance Monitoring:

Set up Google Play Console and App Store Connect to monitor crash logs, app performance metrics (load time, responsiveness), and battery consumption.

Use tools like Firebase Analytics to track user engagement, monitor session times, and analyze how users progress through different levels.

3. User Feedback Collection:

Implement a feedback mechanism within the app (post-level surveys, app reviews) to gather user experiences and suggestions for improvement.

Monitor app reviews on the Play Store and App Store for any reported issues or bugs post-launch.

4. Update and Bug Fix Schedule:

Establish a schedule for rolling out patches and updates to fix any post-launch issues.

Hotfixes should be deployed within 24-48 hours for critical bugs.

New content updates (e.g., new standards, levels, or challenges) can be planned for monthly or quarterly releases.

5. Backup and Recovery:

Ensure regular backups of user data (progress, achievements, profiles) are happening on the cloud (Firebase or other services).

Implement disaster recovery strategies to quickly restore the app to a previous state if any major issues arise.

6. Security Validation:

Validate that user data (e.g., profile info, scores) is encrypted during transit and stored securely.

Review server-side and database security configurations to prevent unauthorized access or data breaches.

7. Marketing & User Engagement:

Promote the app's availability through social media, email campaigns, and relevant forums.

Use push notifications to inform users about new features, updates, and upcoming levels, ensuring user engagement post-launch.

By following this deployment process and checking off the necessary tasks post-launch, the game app will be successfully delivered, maintained, and continually improved, ensuring both engagement and educational impact.

User Guide

a. Installation Instructions

For Android (Google Play Store):

1. Open the Google Play Store on your Android device.

2. In the search bar, type the name of the app (e.g., "GET STANDARDIZED").

3. Locate the app in the search results and tap on it.

4. Tap Install to download and install the app on your device.

5. Once the installation is complete, tap Open to launch the game.

For iOS (Apple App Store):

1. Open the App Store on your iOS device.

2. Use the search bar to find the app (e.g., "BIS Awareness Game").

3. Tap on the app in the search results.

4. Tap Get to download and install the app.

5. Once installed, tap Open to start the game.

b. User Interface Overview

1. Home Screen:

Displays available game modes (e.g., Levels, Quizzes, Challenges).

Access to Profile, Leaderboard, Settings, and Rewards.

Shows your current progress and total score.

2. Main Menu:

Play Game: Takes you to the level or challenge selector.

Leaderboard: View global rankings based on user scores.[Future Plan]

Profile: Edit your user profile and view achievements.

Settings: Adjust sound, notifications, and other preferences.

3. Level/Challenge Screen:

Displays a list of available levels and challenges.

Shows locked levels (if any) with information on how to unlock them.

4. Question Interface:

Question Area: Displays the current quiz question.

Answer Options: A list of possible answers in multiple-choice format.

Timer: A countdown timer to answer each question.

Submit Button: Confirms your answer and moves to the next question.

5. Rewards Screen:

View badges and achievements you have earned.

Shows requirements for unlocking future badges.

6. Settings Screen:

Sound Settings: Adjust or mute music and sound effects.

Notification Settings: Enable/disable push notifications for new levels, updates, etc.

Language Preferences: Select your preferred language.

c. How to Use the Software

1. Starting a Game:

After installation, open the app and register or log in with your account.

Once logged in, the home screen will display your progress, current score, and unlocked levels.

Tap on Play Game to select a level or challenge.

Start with the beginner levels to learn the basics of BIS standards.

2. Playing a Level:

Each level contains educational content about specific BIS standards, followed by a series of quizzes or challenges.

Follow the on-screen prompts to complete each level.

Answer quiz questions to progress through the game. Your score is based on the accuracy and speed of your responses.

Upon completing a level, you will earn points, and possibly unlock badges or new levels.

3. Earning Rewards:

Complete levels and questions to earn points.

As you accumulate points, you'll unlock badges for specific achievements, like completing a certain number of levels or scoring above a threshold.

You can view your badges and certificates in the Rewards section of the app.

4. Viewing Leaderboards:

Go to the Leaderboard section from the main menu to see how you rank globally or against your friends.

Compare scores, and aim to reach the top of the leaderboard by improving your quiz and challenge results.

5. Managing Your Profile:

Access your profile from the home screen.

You can update your username, view your achievements, and track your progress across various levels.

6. Updating Settings:

Adjust sound, notification preferences, and language in the Settings section.

You can also check for app updates here.

Maintenance and Support

a. Troubleshooting Guide

1. Login Issues:

Problem: Unable to log in to the app.

Solution: Ensure you are using the correct credentials. If you've forgotten your password, use the Forgot Password option to reset it.

Tip: Check your internet connection and try again.

2. App Crashes or Freezes:

Problem: The app crashes or freezes while playing.

Solution:

1. Close the app and restart it.

2. Ensure your device's software is up to date.

3. Clear the app cache (for Android) or reinstall the app.

Tip: If the issue persists, contact support via the app or website.

3. Slow Performance:

Problem: The app runs slowly or lags during gameplay.

Solution:

1. Close any other apps running in the background.

2. Check your device's available storage and clear space if necessary.

3. Lower graphic settings in the Settings menu (if available).

Tip: Ensure your device meets the minimum system requirements.

4. Questions/task Won’t Load:

Problem: A level or quiz fails to load.

Solution:

1. Check your internet connection.

2. Restart the app and attempt to load the quiz again.

Tip: If this occurs frequently, report the issue using the app’s support feature.

5. Missing Progress or Rewards:

Problem: Points or badges are not showing after completing a level.

Solution:

1. Log out and log back in to refresh your progress.

2. Check the Rewards section to ensure they are not delayed.

Tip: Contact support if the issue is not resolved after a refresh.

6. Push Notifications Not Working:

Problem: Not receiving notifications about new levels or updates.

Solution:

1. Check if notifications are enabled in the Settings menu.

2. Ensure app notifications are enabled in your device's settings.

Tip: Restart the app to see if notifications resume.

7. Update Issues:

Problem: Unable to update the app from the Play Store or App Store.

Solution:

1. Ensure you have a stable internet connection.

2. Check if your device has enough storage space for the update.

Tip: If you still cannot update, try uninstalling and reinstalling the app.

By following the steps outlined in this guide, users will have a smooth experience with the BIS Awareness Game app, from installation to gameplay, while addressing any technical issues that may arise.

Appendices

a. Glossary of Terms

1. BIS (Bureau of Indian Standards): The national standards body of India responsible for the development and management of standards in various industries.

2. API (Application Programming Interface): A set of tools and protocols used for building and interacting with software applications.

3. CI/CD (Continuous Integration/Continuous Deployment): Practices in software development that enable automatic code testing and deployment.

4. APK (Android Package Kit): The file format used for distributing and installing apps on Android devices.

5. Firebase: A platform developed by Google for building mobile and web applications, offering backend services such as database, authentication, and cloud storage.

b. References

1. BIS Standards Documentation: Available at the BIS official website.

2. Unreal game engine Documentation: Official unreal game engine development platform resources.

3. Firebase Documentation: Resources on using Firebase backend services for mobile apps.

c. Additional Resources

1. Google Play Developer Console: Tools for publishing Android apps.

2. Apple Developer Program: Information for publishing iOS apps.

3. TestFlight: A platform for beta testing iOS apps before public release.

4.Video refrences:
[1] https://bis.gov.in/PDF/pdf/rti/manual.pdf
[2] https://www.bis.gov.in/standards/standards-overview/
[3] https://consumeraffairs.nic.in/organisation-and-units/division/bureau-indian-standards
[4] https://www.bis.gov.in/the-bureau/bis-act-rules-and-regulations/
[5] https://www.bis.gov.in/standards/technical-information-services/
[6] https://www.bis.gov.in
[7]https://www.bis.gov.in/product-certification/product-specific-information-2/product-manualsmk/
[8] https://law.resource.org/pub/in/bis/S07/is.2381.2.2009.pdf

d. Mandatory Declarations

We, the undersigned, declare that:

1. The game app submitted to the Bureau of Indian Standards Hackathon is our original work.

2. All team members have contributed to the development of this project.

3. No part of the project infringes upon any third-party intellectual property rights.

Rutheeshtaa Su
Akshay V
Sudarshan M
Aswanth M
Gayathri R