**Logo, company name

Description automatically generated**

**COMSATS University Islamabad (CUI)**

Software Design Description   
(SDS DOCUMENT)

for

**BrainBee**

Version 1.0

***By***

**Nasir Shahzad CUI/SP22-BCS-077/ISB**

**Sohail Faiz Rasool CUI/SP22-BCS-095/ISB**

***Supervisor*Mr. Tanveer Ahmed**

*Bachelor of Science in Computer Science (2022-2026)*

Table of Contents

[**1.** **Introduction** 2](#_Toc199007893)

[**1.1** **Modules** 2](#_Toc199007894)

[**1.1.1** **Student App** 2](#_Toc199007895)

[**1.1.2** **Parent App** 3](#_Toc199007896)

[**1.1.3** **Teacher App** 3](#_Toc199007897)

[**2.** **Design Methodology and Software Process Model** 3](#_Toc199007898)

[**2.1** **Design Methodology** 3](#_Toc199007899)

[**2.2** **Software Process Model** 4](#_Toc199007900)

[**3.** **System Overview** 4](#_Toc199007901)

[**3.1** **Functionality** 5](#_Toc199007902)

[**3.2** **Context** 5](#_Toc199007903)

[**3.3** **Design** 5](#_Toc199007904)

[**3.4** **Architectural Design** 6](#_Toc199007905)

[**4.** **Design Models** 6](#_Toc199007906)

[**5.** **Data Design** 6](#_Toc199007907)

[**5.1** **Data Dictionary** 6](#_Toc199007908)

[**5.2** **Data Schema** 15](#_Toc199007909)

[**5.2.1** **User Schema (Base Schema)** 15](#_Toc199007910)

[**5.2.2** **Student Schema (Extends User)** 16](#_Toc199007911)

[**5.2.3** **Parent Schema (Extends User)** 16](#_Toc199007912)

[**5.2.4** **Teacher Schema (Extends User)** 17](#_Toc199007913)

[**5.2.5** **Quiz Schema** 17](#_Toc199007914)

[**5.2.6** **Battle Schema** 18](#_Toc199007915)

[**5.2.7** **FlashCard Schema** 18](#_Toc199007916)

[**5.2.8** **Assignment Schema** 19](#_Toc199007917)

[**5.2.9** **Reward Schema** 19](#_Toc199007918)

[**5.2.10** **Goal Schema** 19](#_Toc199007919)

[**5.2.11** **Report Card Schema** 20](#_Toc199007920)

[**5.2.12** **Chat History Schema** 20](#_Toc199007921)

[**5.2.13** **Class Schema** 21](#_Toc199007922)

[**6.** **Human Interface Design** 22](#_Toc199007923)

[**6.1** **Screen Images** 22](#_Toc199007924)

[**6.1.1** **Login screen** 23](#_Toc199007925)

[**6.1.2** **SignUp screen** 24](#_Toc199007926)

[**6.1.3**  **Dashboard** 25](#_Toc199007927)

[**6.1.4** **Goals Screen** 26](#_Toc199007928)

[**6.1.5**  **Battle Screen** 27](#_Toc199007929)

[**6.1.6**  **Select Subject for Battle:** 28](#_Toc199007930)

[**6.1.7** **Select Chapter For Battle:** 29](#_Toc199007931)

[**6.1.8** **Quiz Battle Screen:** 30](#_Toc199007932)

[**6.1.9** **Result Screen:** 31](#_Toc199007933)

[**6.1.10**  **Battle Report Card:** 32](#_Toc199007934)

[**6.1.11**  **Battle invitation Screen:** 33](#_Toc199007935)

[**6.1.12** **Enrolled Classes:** 34](#_Toc199007936)

[**6.1.13**  **Class Detail Screen:** 35](#_Toc199007937)

[**6.1.14**  **Overall Score screen of in app Quizzes:** 36](#_Toc199007938)

[**6.1.15** **Book wise Score:** 37](#_Toc199007939)

[**6.1.16** **LeaderBoard:** 38](#_Toc199007940)

[**6.1.17** **Menu:** 39](#_Toc199007941)

[**6.1.18** **Notifications:** 40](#_Toc199007942)

[**6.2** **Screen Objects and Actions** 41](#_Toc199007943)

[**6.2.1** **Login Screen:** 41](#_Toc199007944)

[**6.2.2** **Sign Up Screen:** 41](#_Toc199007945)

[**6.2.3** **Home/Dashboard Screen:** 41](#_Toc199007946)

[**6.2.4** **Battle Overview Screen:** 41](#_Toc199007947)

[**6.2.5** **Select Subject Screen (for Battle):** 41](#_Toc199007948)

[**6.2.6** **Select Chapters Screen:** 41](#_Toc199007949)

[**6.2.7** **Brain Battle Screen:** 41](#_Toc199007950)

[**6.2.8** **Battle Result Popup:** 41](#_Toc199007951)

[**7.** **Implementation** 43](#_Toc199007952)

[**7.1** **Algorithms** 43](#_Toc199007953)

[**7.1.1** **Get Flashcards** 43](#_Toc199007954)

[**7.1.2** **Match Peer For Battle** 43](#_Toc199007955)

[**7.1.3** **Generate Quiz** 44](#_Toc199007956)

[**7.1.4** **Generate Notes From Video** 44](#_Toc199007957)

[**7.1.5** **Assign Goal** 45](#_Toc199007958)

[**7.1.6** **Enroll In Class** 45](#_Toc199007959)

[**7.1.7** **Calculate Progress** 45](#_Toc199007960)

[**7.1.8** **Check Rewards** 46](#_Toc199007961)

[**7.1.9** **Evaluate Quiz** 46](#_Toc199007962)

[**7.1.10** **Get Parent Dashboard** 47](#_Toc199007963)

[**7.1.11** **Generate AI Flashcards** 47](#_Toc199007964)

[**7.1.12** **Recommend Content** 48](#_Toc199007965)

[**7.1.13** **Match For Battle** 48](#_Toc199007966)

[**7.1.14** **Calculate Overall Score** 49](#_Toc199007967)

[**7.1.15** **Calculate Book-Wise Scores** 49](#_Toc199007968)

[**7.1.16** **Recommend Content (Hybrid)** 50](#_Toc199007969)

[**7.1.17** **Generate Explanation** 50](#_Toc199007970)

[**7.1.18** **Generate Summary** 51](#_Toc199007971)

[**7.1.19** **Check and Reward Goal** 51](#_Toc199007972)

[**7.2** **External APIs/SDKs** 52](#_Toc199007973)

[**8.** **Testing and Evaluation** 53](#_Toc199007974)

[**8.1** **Unit testing:** 53](#_Toc199007975)

[**8.2** **Functional testing:** 64](#_Toc199007976)

[**8.3** **Business Rules Testing:** 108](#_Toc199007977)

[**8.4** **Integration Testing:** 132](#_Toc199007978)

# **Introduction**

The BrainBee project aims to develop a mobile-based, AI-powered learning platform tailored for Pakistani students in grades 5–12. The system is designed to address gaps in accessibility, engagement, and personalized support in the education sector by leveraging artificial intelligence and gamification. The goal is to make high-quality education affordable, engaging, and effective for students in urban and rural regions alike.

## **Modules**

This subsection lists the modules of BrainBee, along with their functionalities.

### **Student App**

#### **User Management**

Manages user registration, profile updates, account deletion, grade selection, subject preferences, and premium upgrades.

#### **Peer Competition Module**

Enables students to participate in AI-generated quiz battles, track performance, maintain battle history, and view leaderboard positions.

#### **Learning Module**

Provides AI-generated summaries and explanations, interactive chatbot support, and personalized flashcards.

#### **Assessment Module**

Generates personalized quizzes, enables quiz and assignment submission, provides instant feedback, and maintains report cards and leaderboard metrics.

#### **Achievement and Rewards Module**

Tracks and displays badges, certificates, redeemable coins, and coin quests to gamify the learning experience.

#### **Performance Analytics Module**

Offers detailed performance reports across books, report cards, and leaderboard standings over time.

#### **AI Powered Assistant Module**

Allows students to ask questions and receive AI-generated answers with chat history maintenance.

### **Parent App**

#### **User Management**

Manages parent user accounts including registration, profile updates, and deletion.

#### **Monitor Children**

Allows parents to add/remove children, monitor performance, view leaderboard rankings, and receive reports and notifications.

#### **Goals Module**

Enables parents to set, track, and reward academic goals for their children.

#### **Communication Module**

Facilitates chat with teachers and feedback submission to admins.

### **Teacher App**

#### **User Management**

Enables account management for teachers, including registration, profile updates, and deletion.

#### **Class Management**

Allows creation and management of classes, students, study material, assignments, and quizzes.

#### **Assessment Module**

Provides performance monitoring and insights through ML algorithms, auto-grading of assignments, and personalized learning path generation.

# **Design Methodology and Software Process Model**

This section outlines the chosen design methodology and software process model for the BrainBee educational application.

## **Design Methodology**

BrainBee employs **Object-Oriented Programming (OOP)** as its design methodology due to the modular and scalable nature of the application. OOP provides a structured way to represent complex real-world entities such as students, parents, teachers, quizzes, rewards, and performance analytics through classes and objects. This approach ensures high code **reusability**, **maintainability**, and **extensibility**, which is essential for a system that involves multiple interacting modules and diverse user types.

Encapsulation and abstraction enable developers to isolate features like assessments, goal tracking, and peer competition, making the application more manageable and easier to debug. OOP's use of inheritance and polymorphism also supports future feature expansions without major codebase refactoring. This design methodology aligns well with BrainBee’s goal of building a flexible and robust platform that supports continuous improvement and evolving educational needs.

## **Software Process Model**

BrainBee follows the **Agile Software Development** process model due to its iterative, collaborative, and adaptive nature. Agile is particularly well-suited for BrainBee because the project involves AI integration, dynamic feedback from students, teachers, and parents, and frequent improvements based on user behavior and evolving academic needs.

Agile allows the development team to work in **sprints**, delivering incremental updates with fully functional modules such as the Student Quiz Module, AI Chatbot, or Parent Monitoring Dashboard. Regular feedback from stakeholders ensures that any required changes are identified early and implemented quickly. This reduces risk, enhances product quality, and improves alignment with end-user expectations. Agile’s emphasis on working software, adaptability, and close collaboration makes it ideal for an education-focused application aiming for continuous improvement and user engagement.

This figure illustrates the agile development process model and its stages.

A diagram of a software development process

AI-generated content may be incorrect.

Figure 1 Agile Development Model

# **System Overview**

This section provides a high-level summary of the overall system architecture and functionalities of **BrainBee**.

## **Functionality**

**BrainBee** is an AI-powered mobile educational platform designed to enhance academic learning for students in grades 5 to 12 across Pakistan. It offers features such as user management for students, parents, and teachers; personalized learning content; AI-generated quizzes and explanations; gamified peer competition; performance analytics; and a reward system. The platform also provides tools for parental monitoring, goal setting, and teacher-based class and assignment management. Real-time feedback and AI tutoring ensure a customized, engaging, and efficient learning experience. The system promotes academic success through individualized learning paths, gamification, and collaborative support among users.

## **Context**

Operating within the educational ecosystem of Pakistan, **BrainBee** addresses key challenges such as the lack of personalized instruction, rote learning, absence of student motivation, and limited parental and teacher engagement. The platform integrates artificial intelligence to personalize study material and assessments while encouraging healthy competition and progress tracking. By supporting students, parents, and teachers, BrainBee bridges the educational gap between urban elite and under-resourced communities, especially in Punjab. It aims to make quality education accessible, affordable, and interactive through AI and mobile technology.

## **Design**

**BrainBee** utilizes a **multi-tiered architecture** that separates the system into distinct layers:

* **Presentation Layer** for user interfaces across mobile apps (Flutter-based)
* **Business Logic Layer** for managing core functionalities such as quizzes, performance tracking, and reward systems
* **Data Access Layer** for interacting with the MongoDB database securely

This design promotes **modularity**, **reusability**, and **scalability**, allowing different modules (e.g., student assessments, parent goals, teacher analytics) to work independently yet cohesively. RESTful APIs connect the layers to ensure seamless communication across different system components and user apps. The AI components are integrated through Python-based machine learning models that assist in generating personalized content, performance predictions, and quiz recommendations.

## **Architectural Design**

**BrainBee** is designed with a modular architecture to support maintainability, extensibility, and integration of new features. The system is divided into key modules:

* **User Management:** Handles registration, profile management, and user type switching
* **Assessment Module:** Manages AI-generated quizzes, assignments, feedback, and report cards
* **Learning Module:** Provides summaries, flashcards, and AI chatbot explanations
* **Peer Competition Module:** Enables quiz battles, leaderboard tracking, and win/loss history
* **Performance Analytics:** Tracks detailed academic data and AI-based progress insights
* **Parent & Teacher Modules:** Allow monitoring, goal setting, and personalized teaching interventions
* **Reward Module:** Handles badges, coins, certificates, and redemption tracking

This modular approach ensures that updates or changes in one part of the system do not affect others. The **multi-tiered architectural style** supports separation of concerns and eases testing, deployment, and scaling.

# **Design Models**

# **Data Design**

This section outlines the structure and organization of the data used in **BrainBee**, detailing the schema, data models, and relationships essential for efficient data management and retrieval.

## **Data Dictionary**

These tables show the data dictionary used in the **BrainBee** system.

**User (Common Base for Student, Parent, Teacher):**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique user identifier |
| name | String | Full name of the user |
| email | String | Registered email address |
| passwordHash | String | Hashed password |
| userType | Enum | STUDENT, PARENT, or TEACHER |
| profilePhotoUrl | String? | URL to profile image |
| createdAt | DateTime | Account creation timestamp |
| updatedAt | DateTime | Last account update timestamp |

**Student (extends User):**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| grade | Int | Grade level of the student |
| subjects | List | Subjects enrolled |
| isPremium | Boolean | Premium membership status |
| friends | List | Friends list |
| battleHistory | List | History of quiz battles |
| quizResults | List | Quiz attempt records |
| flashcards | List | Flashcards created by the student |
| reportCards | List | Academic report cards |
| badges | List | Earned badges |
| certificates | List | Awarded certificates |
| coins | Int | Current coin balance |
| coinQuests | List | Active and completed coin quests |
| streakScore | Int | Streak count for daily activity |
| dailyLives | Int | Remaining lives for daily quizzes |
| chatHistory | List | Chat history |

**Parent (extends User):**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| children | List | Associated child accounts |
| Monitored  -ReportCards | List | Report cards being monitored |
| notifications | List | Received notifications |
| goals | List | Goals set for children |

**Teacher (extends User):**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| classes | List | Assigned classes |
| materials | List | Teaching materials |
| assignments | List | Given assignments |
| quizTemplates | List | Quiz templates created |
| performanceInsights | List | Performance analytics |

**Subject:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique subject ID |
| name | String | Name of the subject |
| book | Book | Associated book |

**Book:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique book ID |
| title | String | Title of the book |
| chapters | List | Chapters in the book |

**Chapter:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique chapter ID |
| title | String | Title of the chapter |
| contentUrl | String | URL to content |

**Quiz:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique quiz ID |
| creator | User? | Creator of the quiz (or AI) |
| questions | List | List of questions |
| topicTags | List | Related topics |
| createdAt | DateTime | Quiz creation date |

**Question:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique question ID |
| text | String | Question text |
| choices | List | Answer options |
| correctAnswerIndex | Int | Index of correct choice |

**QuizResult:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique result ID |
| student | Student | Student who took the quiz |
| quiz | Quiz | Quiz taken |
| answers | List | Student answers |
| correctCount | Int | Number of correct answers |
| incorrectCount | Int | Number of incorrect answers |
| timestamp | DateTime | When the quiz was taken |

**Answer:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| question | Question | Associated question |
| selectedIndex | Int | Selected answer index |
| isCorrect | Boolean | Whether the answer was correct |
| responseTimeMs | Int | Time taken to answer (ms) |

**Battle:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique battle ID |
| challenger | Student | Initiating student |
| opponent | Student | Opponent student |
| quiz | Quiz | Quiz used for battle |
| challengerScore | Int | Score of challenger |
| opponentScore | Int | Score of opponent |
| winner | Student? | Winner of the battle |
| timestamp | DateTime | Battle timestamp |

**ReportCard:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique report card ID |
| student | Student | Associated student |
| period | Enum | WEEKLY, MONTHLY, or YEARLY |
| scoresByBook | Map<Book, Float> | Performance by book |
| generatedAt | DateTime | Date of report generation |

**Flashcard:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique flashcard ID |
| student | Student | Owner of the flashcard |
| question | String | Flashcard question |
| answer | String | Flashcard answer |
| chapter | Chapter | Linked chapter |

**ChatMessage:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique message ID |
| sender | User | Message sender |
| recipient | User | AIModel | Message recipient |
| content | String | Message content |
| timestamp | DateTime | Time of sending |

**CoinQuest:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique quest ID |
| title | String | Quest title |
| description | String | Quest description |
| coinReward | Int | Coins awarded |
| isCompleted | Boolean | Completion status |
| assignedAt | DateTime | Assignment date |
| completedAt | DateTime? | Completion date |

**Badge:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique badge ID |
| name | String | Name of the badge |
| iconUrl | String | Badge icon URL |
| criteria | String | Criteria to earn the badge |

**Certificate:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique certificate ID |
| student | Student | Certificate holder |
| title | String | Certificate title |
| issuedAt | DateTime | Issue date |
| url | String? | Optional certificate link |

**Notification:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique notification ID |
| recipient | User | Notification recipient |
| type | Enum | QUIZ\_COMPLETED, GOAL\_PROGRESS, etc. |
| payload | JSON | Additional data |
| isRead | Boolean | Read status |
| createdAt | DateTime | Notification timestamp |

**Goal:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique goal ID |
| parent | Parent | Goal setter |
| child | Student | Goal target |
| description | String | Goal description |
| rewardCoins | Int | Reward coins |
| progress | Float | Progress (0.0–1.0) |
| status | Enum | PENDING or COMPLETED |

**Classroom:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique classroom ID |
| teacher | Teacher | Class teacher |
| name | String | Class name |
| students | List | Enrolled students |
| materials | List | Teaching materials |

**Material:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique material ID |
| classroom | Classroom | Related classroom |
| title | String | Material title |
| contentUrl | String | Content file URL |
| uploadedAt | DateTime | Upload time |

**Assignment:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique assignment ID |
| classroom | Classroom | Related class |
| title | String | Assignment title |
| description | String | Details about the assignment |
| dueDate | DateTime | Submission deadline |
| submissions | List | Student submissions |

**Submission:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique submission ID |
| assignment | Assignment | Related assignment |
| student | Student | Student who submitted |
| fileUrl | String | File link |
| submittedAt | DateTime | Time of submission |
| grade | Float? | Assigned grade |
| feedback | String? | Teacher's feedback |

**QuizTemplate:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique template ID |
| classroom | Classroom | Linked classroom |
| title | String | Template name |
| templateData | JSON | Quiz blueprint |

**PerformanceInsight:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| id | UUID | Unique insight ID |
| teacher | Teacher | Associated teacher |
| student | Student | Student being analyzed |
| book | Book | Targeted book |
| metrics | JSON | Metrics like difficulty and timing |
| generatedAt | DateTime | Timestamp of generation |

## **Data Schema**

### **User Schema (Base Schema)**

const UserSchema = new mongoose.Schema({

userID: { type: String, unique: true, required: true },

fullName: String,

email: { type: String, unique: true, required: true },

phoneNumber: String,

passwordHash: String,

role: { type: String, enum: ['Student', 'Parent', 'Teacher'], required: true },

accountType: { type: String, enum: ['Standard', 'Premium'], default: 'Standard' },

profilePicture: String,

registrationDate: { type: Date, default: Date.now },

isActive: { type: Boolean, default: true }

});

### **Student Schema (Extends User)**

const StudentSchema = new mongoose.Schema({

userID: { type: String, ref: 'User' },

gradeLevel: String,

enrolledSubjects: [String],

quizHistory: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Quiz' }],

leaderboardRank: Number,

streakScore: Number

});

### **Parent Schema (Extends User)**

const ParentSchema = new mongoose.Schema({

userID: { type: String, ref: 'User' },

childrenIDs: [{ type: String, ref: 'Student' }],

notifications: [String]

});

### **Teacher Schema (Extends User)**

const TeacherSchema = new mongoose.Schema({

userID: { type: String, ref: 'User' },

classIDs: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Class' }],

assignedQuizzes: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Quiz' }],

uploadedMaterials: [String]

});

### **Quiz Schema**

const QuizSchema = new mongoose.Schema({

quizID: { type: String, unique: true },

createdBy: String,

subject: String,

topic: String,

questions: [

{

questionText: String,

options: [String],

correctAnswer: String

}

],

generatedByAI: { type: Boolean, default: true },

attemptHistory: [{ studentID: String, score: Number, attemptedAt: Date }],

dateCreated: { type: Date, default: Date.now }

});

### **Battle Schema**

const BattleSchema = new mongoose.Schema({

battleID: { type: String, unique: true },

challengerID: String,

opponentID: String,

quizData: { type: Object }, // Could embed quiz structure

winnerID: String,

battleDate: Date,

durationSeconds: Number,

winLossRecord: {

[String]: { type: String, enum: ['Win', 'Loss', 'Draw'] }

}

});

### **FlashCard Schema**

const FlashCardSchema = new mongoose.Schema({

cardID: { type: String, unique: true },

studentID: String,

subject: String,

topic: String,

content: String,

createdAt: { type: Date, default: Date.now }

});

### **Assignment Schema**

const AssignmentSchema = new mongoose.Schema({

assignmentID: { type: String, unique: true },

classID: String,

uploadedBy: String,

description: String,

deadline: Date,

submissions: [{

studentID: String,

fileURL: String,

status: { type: String, enum: ['Pending', 'Submitted', 'Graded'] }

}]

});

### **Reward Schema**

const RewardSchema = new mongoose.Schema({

rewardID: { type: String, unique: true },

type: { type: String, enum: ['Badge', 'Certificate', 'CoinQuest'] },

name: String,

awardedTo: String,

awardedBy: String,

dateAwarded: { type: Date, default: Date.now },

coinValue: Number

});

### **Goal Schema**

const GoalSchema = new mongoose.Schema({

goalID: { type: String, unique: true },

parentID: String,

studentID: String,

description: String,

status: { type: String, enum: ['Pending', 'Completed', 'Failed'], default: 'Pending' },

rewardCoins: Number,

dueDate: Date

});

### **Report Card Schema**

const ReportCardSchema = new mongoose.Schema({

reportCardID: { type: String, unique: true },

studentID: String,

bookName: String,

score: Number,

weakAreas: [String],

evaluationDate: { type: Date, default: Date.now }

});

### **Chat History Schema**

const ChatHistorySchema = new mongoose.Schema({

chatID: { type: String, unique: true },

studentID: String,

topic: String,

messages: [{

sender: { type: String, enum: ['student', 'AI'] },

text: String,

timestamp: { type: Date, default: Date.now }

}],

createdAt: { type: Date, default: Date.now }

});

### **Class Schema**

const ClassSchema = new mongoose.Schema({

classID: { type: String, unique: true },

teacherID: String,

className: String,

studentList: [{ type: String, ref: 'Student' }],

materials: [{

title: String,

fileURL: String,

uploadedAt: Date

}]

});

# **Human Interface Design**

This section outlines the user interface design principles and components for **BrainBee**, ensuring an intuitive and seamless experience for both mobile app users and admin dashboard operators.

## **Screen Images**

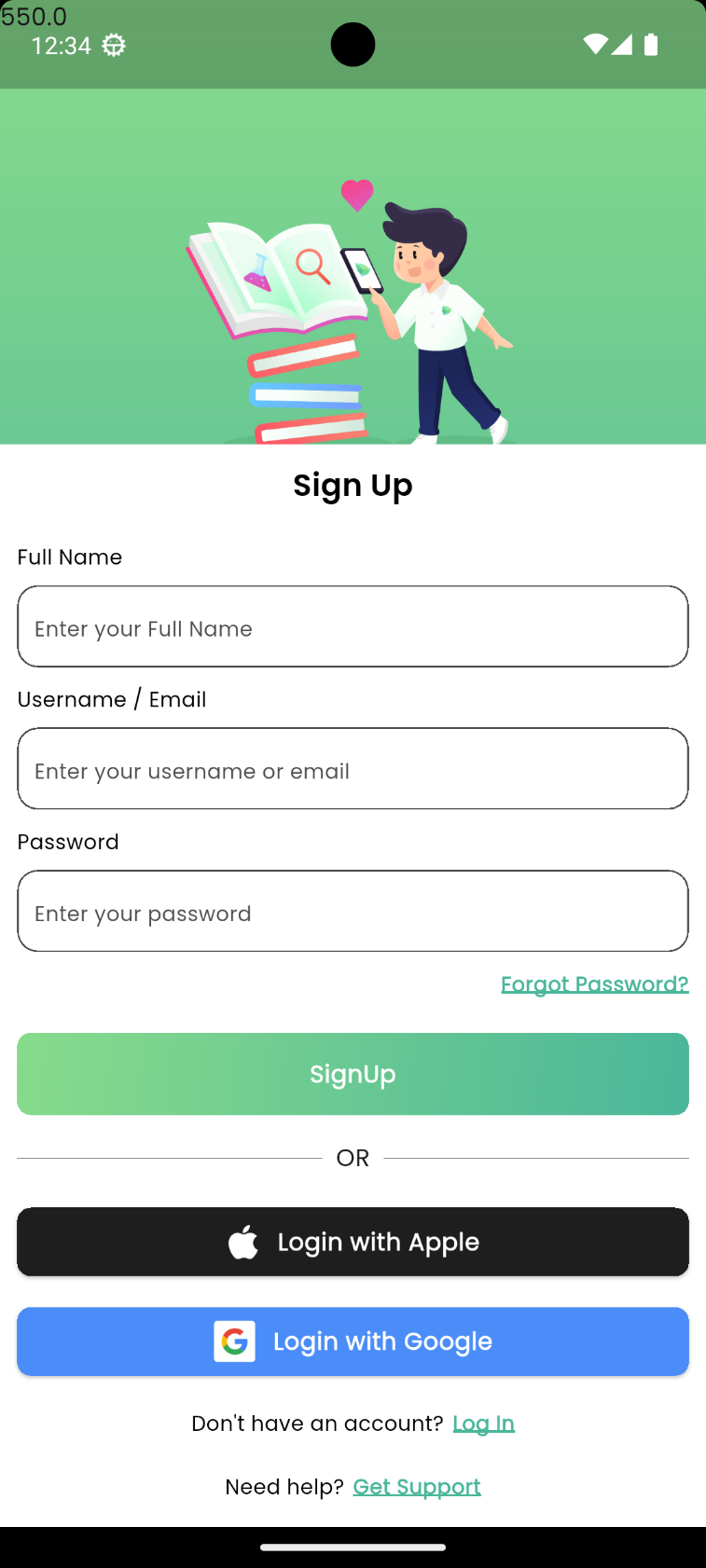
This subsection presents the screen images showing the user interface for the SafarRehnuma platform.

### **Login screen**

**A screenshot of a login form

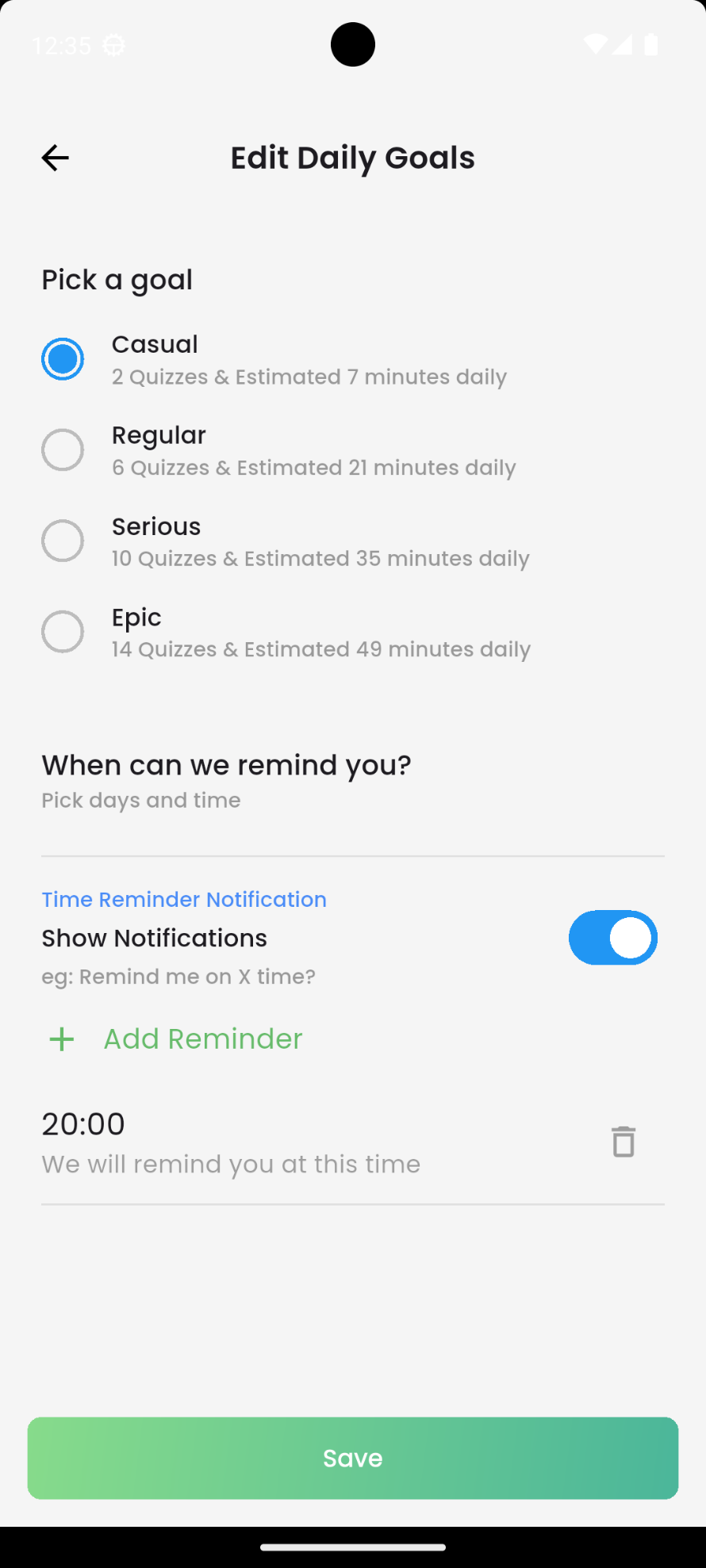
AI-generated content may be incorrect.**

### **SignUp screen**

****

### **Dashboard**

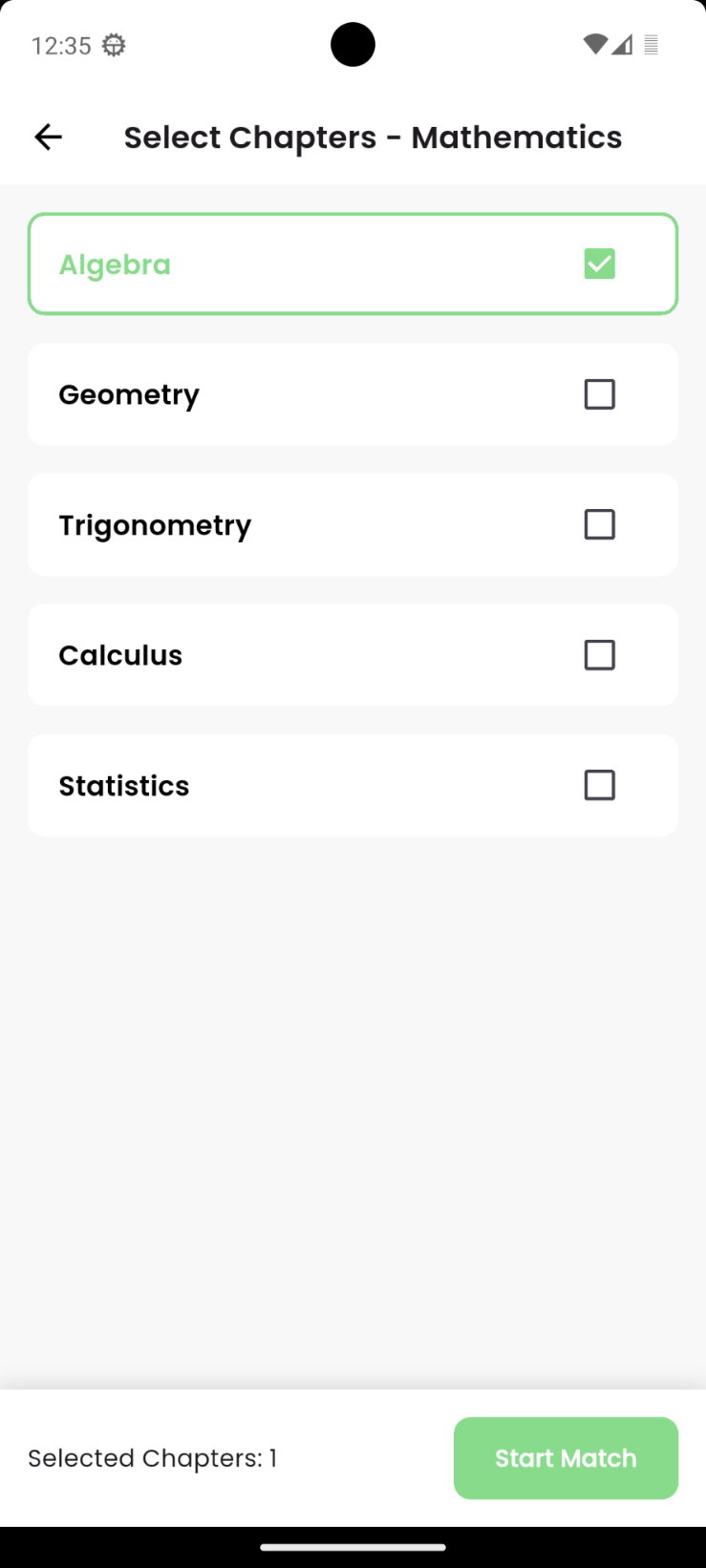
### **Goals Screen**

****

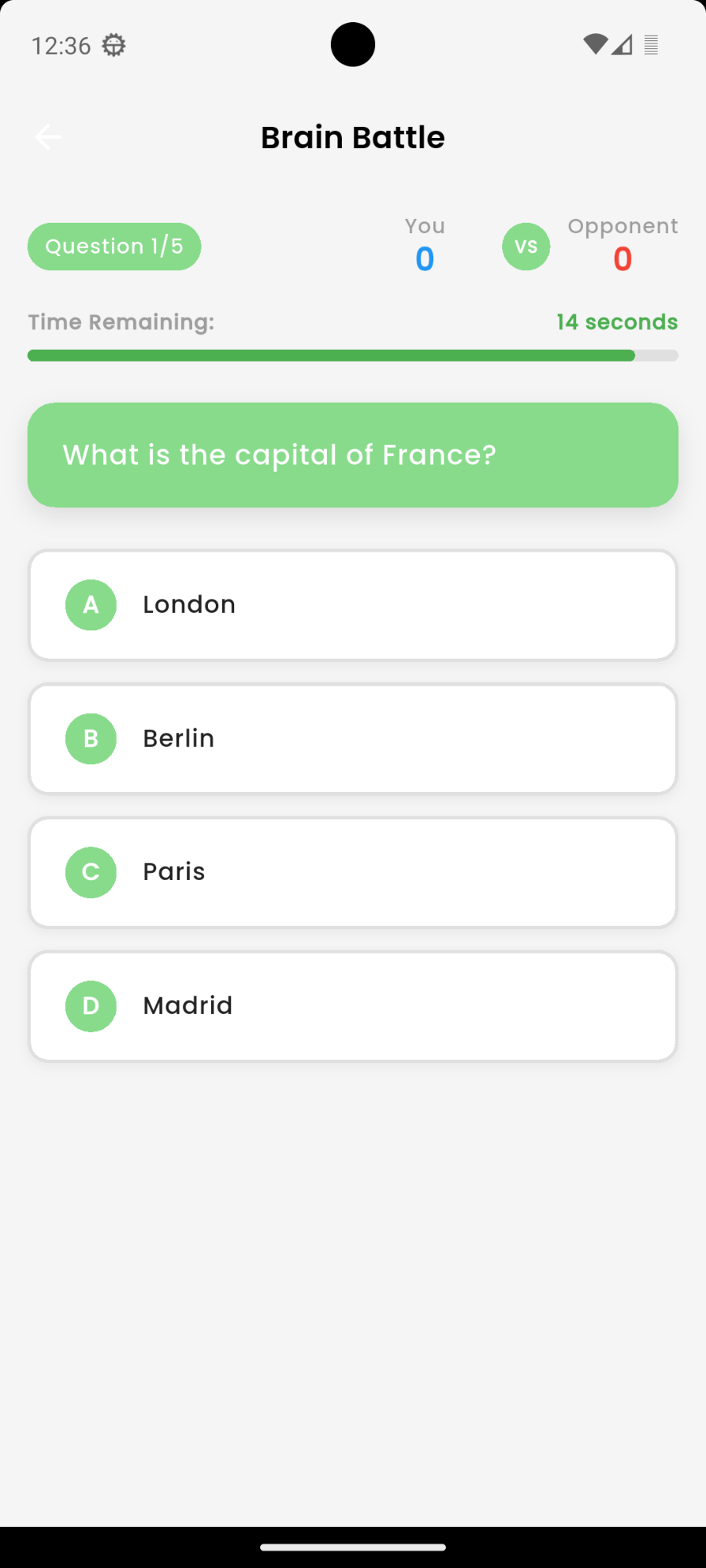
### **Battle Screen**

### **Select Subject for Battle:**

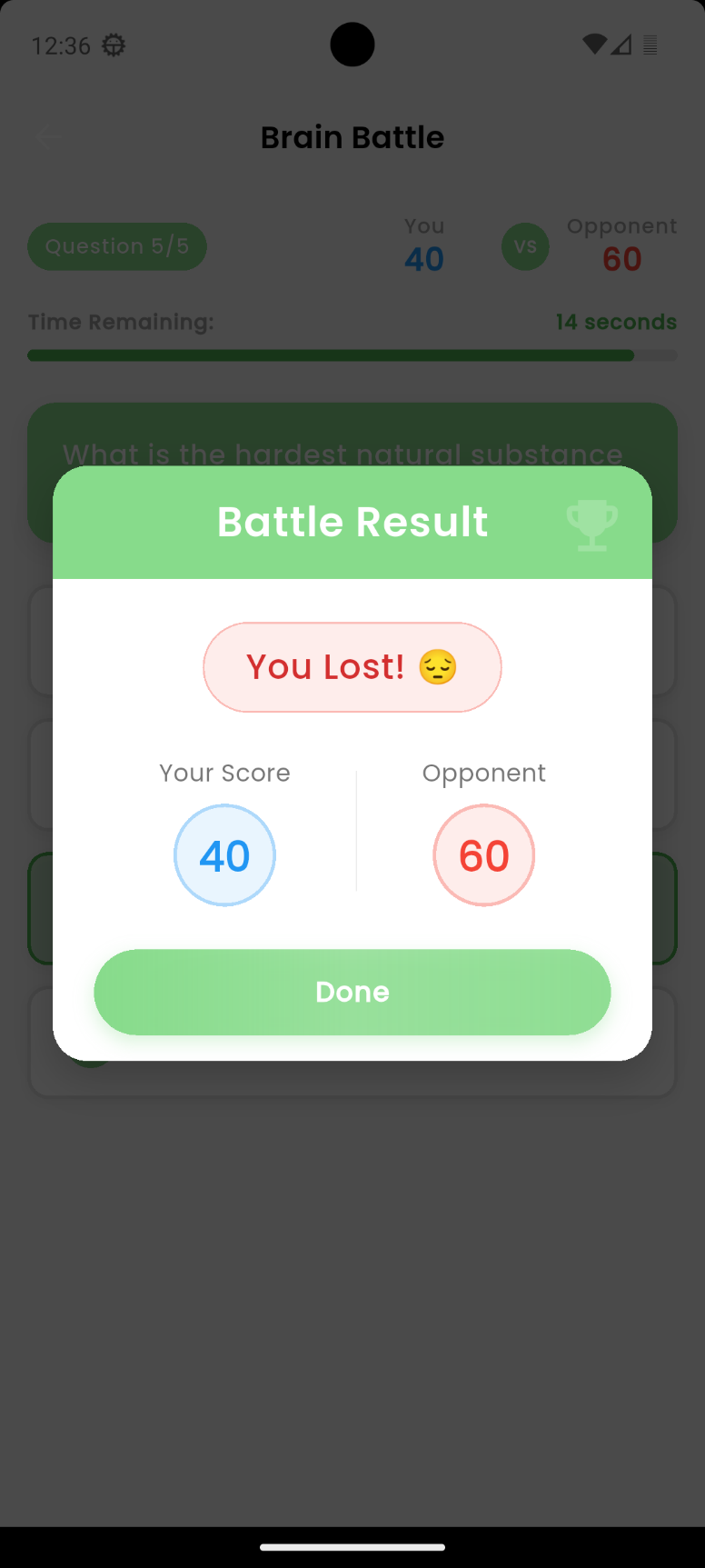
### **Select Chapter For Battle:**

****

### **Quiz Battle Screen:**

****

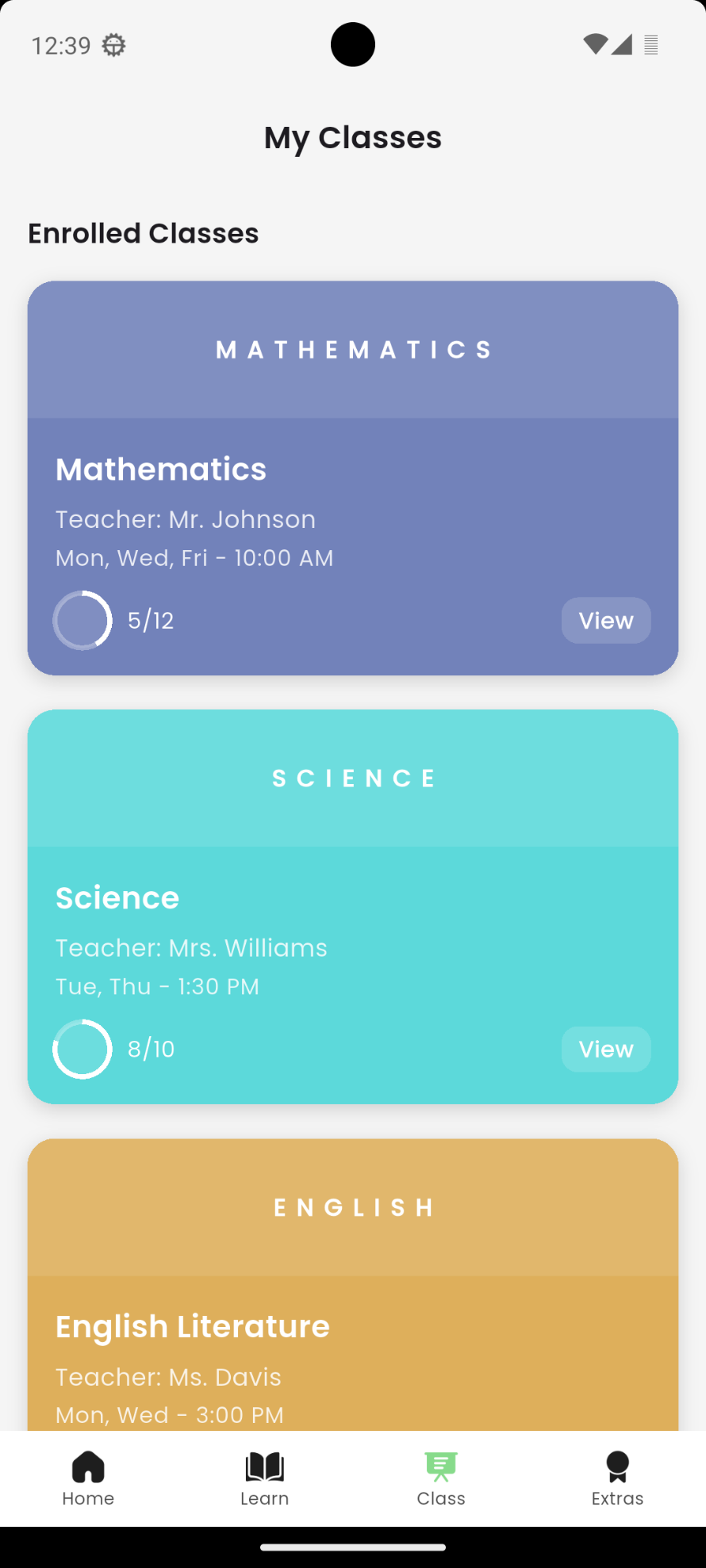
### **Result Screen:**

****

### **Battle Report Card:**

### **Battle invitation Screen:**

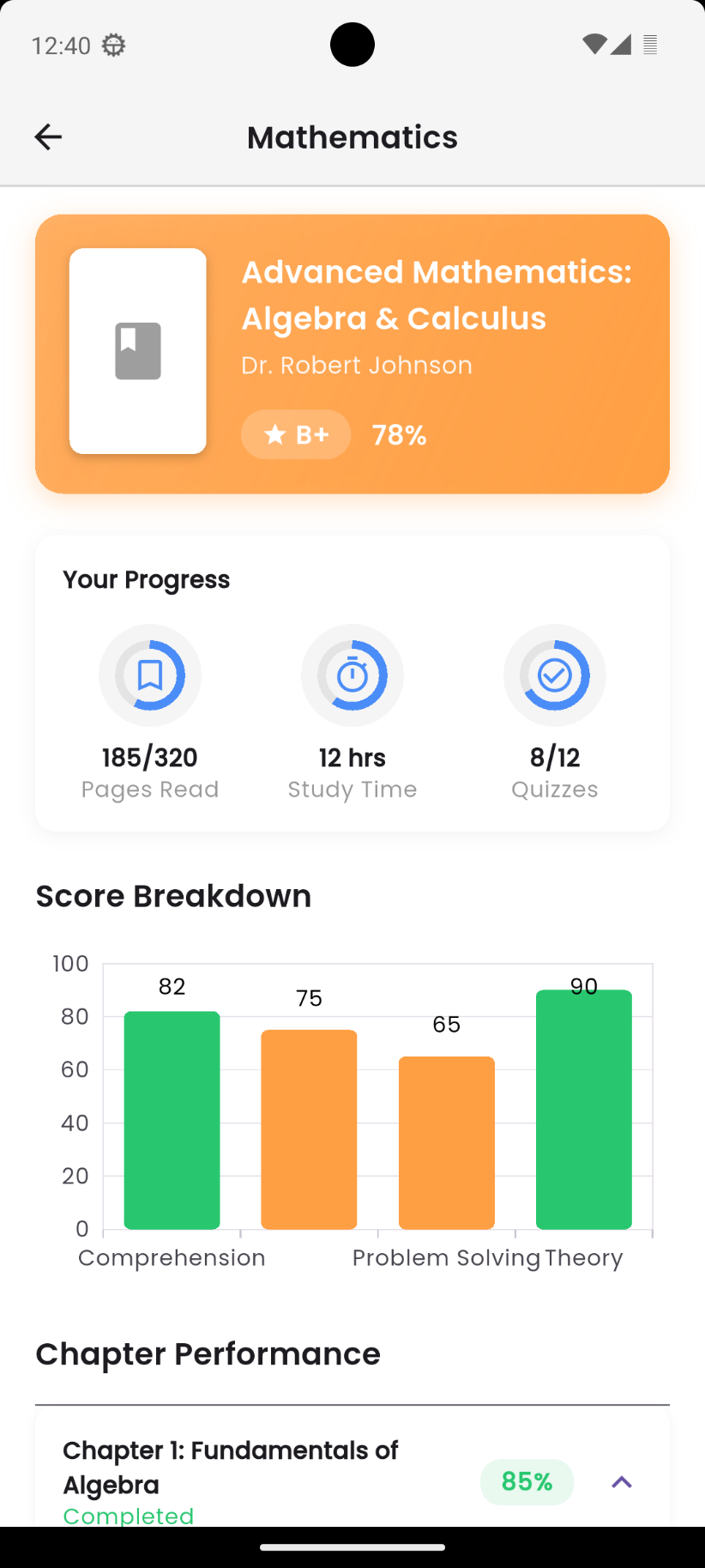
### **Enrolled Classes:**

****

### **Class Detail Screen:**

### **Overall Score screen of in app Quizzes:**

### **Book wise Score:**

****

### **LeaderBoard:**

**A screenshot of a cell phone

AI-generated content may be incorrect.**

### **Menu:**

**A screenshot of a phone

AI-generated content may be incorrect.**

### **Notifications:**

**A screenshot of a chat

AI-generated content may be incorrect.**

## **Screen Objects and Actions**

This subsection outlines the different screens and user interactions present in the **BrainBee** application.

* + 1. **Login Screen:** Allows existing users to access their account using their username/email and password or via social logins (Apple/Google). Provides options for password recovery and new user sign-up.
    2. **Sign Up Screen:** Enables new users to create an account by providing their full name, username/email, and password, or by using social logins. Offers a link to log in for existing users.
    3. **Home/Dashboard Screen:** Displays user's progress (score, achievements), features promotional banners like "Bookmark Questions," and lists available subjects (Mathematics, Physics, Biology) for quizzing. Provides main navigation via bottom tabs.
    4. **Battle Overview Screen:** Shows the user's battle statistics (wins, rank) and a history of past battles. Allows users to start a new battle or join one using an invitation code.
    5. **Select Subject Screen (for Battle):** Presents a list of academic subjects (English, Biology, Mathematics, etc.) for the user to choose from before starting a quiz or battle.
    6. **Select Chapters Screen:** Allows the user to refine their quiz/battle by selecting specific chapters (e.g., Algebra, Geometry) within the previously chosen subject.
    7. **Brain Battle Screen:** Displays the live quiz interface showing the current question, multiple-choice answers, user and opponent scores, question progress, and a countdown timer.
    8. **Battle Result Popup:** A modal dialog that appears after a battle, indicating whether the user won or lost, and showing their score versus the opponent's score.
    9. **Quiz Report Card Screen:** Provides a detailed summary of a completed quiz, including the overall result (Victory!), scores, time taken, accuracy, and a question-by-question analysis of answers.
    10. **Battle Invitation Popup:** Allows a user to join a private battle by entering a 6-digit invitation code obtained from a friend.
    11. **Select Subject Screen (for Flashcards):** Lists available subjects (English, Biology, etc.) and indicates the number of flashcards available for each, allowing users to choose a subject for flashcard study.
    12. **My Classes Screen:** Displays a list of classes the user is enrolled in, showing the subject, teacher, schedule, and progress on assignments for each class.
    13. **Class Details Screen :** Shows detailed information for a specific enrolled class, including schedule, student count, assignment progress, and actions like viewing materials or contacting the teacher.
    14. **Overall Score Screen :** Presents the user's overall academic score and grade, along with statistics (books, quizzes, hours) and a bar chart breaking down performance by subject.
    15. **Subject Details Screen:** Provides an in-depth look at the user's progress within a specific subject (e.g., Advanced Mathematics), including pages read, study time, quiz scores, and a skill-based score breakdown.
    16. **Leaderboard Screen:** Displays rankings of users based on points, with options to view weekly or monthly leaderboards, showcasing top performers and their scores.
    17. **Menu Screen:** Provides access to various account settings (manage account, change password, select grade/subjects), app preferences (language, settings), help, and logout options.
    18. **Notifications Screen:** Lists important updates and alerts for the user, such as earned badges, new goals set by parents, or other app-related information.

# **Implementation**

This section covers the practical implementation aspects of SafarRehnuma, encompassing algorithmic implementations, integration with external interfaces, and the development of user interfaces for seamless interaction.

## **Algorithms**

This subsection explains the algorithms implemented within **BrainBee**, highlighting their roles in optimizing routes, calculating dynamic pricing, and managing ride requests efficiently.

### **Get Flashcards**

| **Input: Book ID, Chapter ID** |
| --- |
| **Output: List of dynamically generated flashcards (question-answer pairs)** |

function GetFlashcards(bookID, chapterID)

chapter ← DB.query("SELECT content FROM Chapters WHERE bookID = ? AND chapterID = ?", bookID, chapterID)

if chapter is null then

throw "Chapter not found"

end if

flashcards ← AI.GenerateFlashcards(chapter.content)

return flashcards

end

### **Match Peer For Battle**

| **Input: Student ID** |
| --- |
| **Output:** Matching result message |

function MatchPeerForBattle(studentID)

waitingList ← GetWaitingPeers()

for each peer in waitingList do

if IsCompatible(peer, studentID) then

CreateBattleRoom(studentID, peer)

return "Matched with " + peer.name

return "No match found. Please wait."

end

### **Generate Quiz**

| **Input: Chapter ID, Difficulty Level** |
| --- |
| **Output:** Quiz (list of questions) |

function GenerateQuiz(chapterID, difficultyLevel)

allQuestions ← GetQuestions(chapterID)

suitable ← FilterByDifficulty(allQuestions, difficultyLevel)

quiz ← RandomSelect(suitable, 10)

return quiz

end

### **Generate Notes From Video**

| **Input: Video ID** |
| --- |
| **Output:** Summarized notes |

function GenerateNotesFromVideo(videoID)

video ← LoadVideo(videoID)

transcript ← Transcribe(video)

notes ← Summarize(transcript)

return notes

end

### **Assign Goal**

| **Input: Child ID, Goal Text, Deadline** |
| --- |
| **Output:** Goal assignment confirmation |

function AssignGoal(childID, goalText, deadline)

goal ← CreateGoal(goalText, deadline)

AssignToChild(childID, goal)

return "Goal assigned"

end

### **Enroll In Class**

| **Input: Student ID, Class ID** |
| --- |
| **Output:** Enrollment status |

function EnrollInClass(studentID, classID)

class ← GetClass(classID)

if class.capacity > class.enrolled then

Enroll(studentID, classID)

return "Enrolled"

return "Class Full"

end

### **Calculate Progress**

| **Input: Student ID, Subject ID** |
| --- |
| **Output:** Progress percentage |

function CalculateProgress(studentID, subjectID)

totalTasks ← GetAllTasks(subjectID)

completed ← GetStudentCompletions(studentID, subjectID)

progress ← (completed / totalTasks) \* 100

return progress

end

### **Check Rewards**

| **Input: Child ID** |
| --- |
| **Output:** Rewards issued (if any) |

function CheckRewards(childID)

goals ← GetGoals(childID)

for goal in goals do

if goal.status == "Completed" and goal.rewardGiven == false then

GiveReward(childID, goal.reward)

MarkRewardGiven(goal)

end

### **Evaluate Quiz**

| **Input: Quiz ID, Student Answers** |
| --- |
| **Output:** Score and feedback |

function EvaluateQuiz(quizID, studentAnswers)

correctAnswers ← GetCorrectAnswers(quizID)

score ← 0

for i in range(len(correctAnswers)) do

if studentAnswers[i] == correctAnswers[i] then

score += 1

feedback ← GenerateFeedback(score)

return score, feedback

end

### **Get Parent Dashboard**

| **Input: Parent ID** |
| --- |
| **Output:** Dashboard data |

function GetParentDashboard(parentID)

children ← GetChildren(parentID)

dashboard ← []

for child in children do

stats ← GetChildStats(child.id)

dashboard.append(stats)

return dashboard

end

### **Generate AI Flashcards**

| **Input: Chapter Text** |
| --- |
| **Output:** Flashcards (Q&A pairs) |

function GenerateAIFlashcards(chapterText)

keyPoints ← ExtractKeyConcepts(chapterText)

flashcards ← []

for point in keyPoints do

question ← GenerateQuestion(point)

answer ← GenerateAnswer(point)

flashcards.append((question, answer))

return flashcards

end

### **Recommend Content**

| **Input: User ID** |
| --- |
| **Output:** Recommended content list |

function RecommendContent(userID)

history ← GetUserHistory(userID)

similarUsers ← FindSimilarUsers(userID)

preferredTopics ← AnalyzeWeakAreas(history)

recommended ← []

for topic in preferredTopics do

content ← FetchContent(topic)

recommended.append(content)

return recommended

end

### **Match For Battle**

| **Input: Student ID** |
| --- |
| **Output:** Matched Peer or Waiting Message |

function MatchForBattle(studentID)

userLevel ← GetUserSkillLevel(studentID)

availablePeers ← GetWaitingStudents()

matchedPeer ← null

for peer in availablePeers do

if abs(GetUserSkillLevel(peer) - userLevel) ≤ threshold then

matchedPeer ← peer

break

if matchedPeer ≠ null then

CreateBattleRoom(studentID, matchedPeer)

return matchedPeer

return "No suitable peer found. Waiting in queue."

end

### **Calculate Overall Score**

| **Input: Student ID** |
| --- |
| **Output:** Overall score percentage |

function CalculateOverallScore(studentID)

books ← GetAllBooksAttempted(studentID)

totalScore ← 0

totalQuestions ← 0

for book in books do

(score, questions) ← GetBookScore(studentID, book.id)

totalScore += score

totalQuestions += questions

if totalQuestions == 0 then

return 0

return (totalScore / totalQuestions) \* 100

end

### **Calculate Book-Wise Scores**

| **Input: Student ID** |
| --- |
| **Output:** Dictionary of book-wise percentages |

function CalculateBookWiseScores(studentID)

books ← GetAllBooksAttempted(studentID)

scores ← {}

for book in books do

(score, questions) ← GetBookScore(studentID, book.id)

if questions > 0 then

percentage ← (score / questions) \* 100

scores[book.title] ← percentage

return scores

end

### **Recommend Content (Hybrid)**

| **Input: User ID** |
| --- |
| **Output:** Ranked list of recommended items |

function RecommendContent(user\_id)

user\_profile ← GetUserPreferences(user\_id)

similar\_users ← FindSimilarUsersUsingCosineSimilarity(user\_profile)

content\_based\_recs ← RecommendBasedOnUserProfile(user\_profile)

collaborative\_recs ← RecommendFromSimilarUsers(similar\_users)

combined\_recs ← MergeAndRank(content\_based\_recs, collaborative\_recs)

return combined\_recs

end

### **Generate Explanation**

| **Input: Topic Text** |
| --- |
| **Output:** Detailed Explanation |

function GenerateExplanation(topic\_text)

keywords ← ExtractKeyConcepts(topic\_text)

explanation ← []

for keyword in keywords do

step\_explanation ← UseGPTOrT5ToExplain(keyword)

explanation.append(step\_explanation)

return explanation

end

### **Generate Summary**

| **Input: Content** |
| --- |
| **Output:** Summary |

function GenerateSummary(content)

cleaned\_text ← PreprocessText(content)

summary ← RunPegasusOrT5Summarizer(cleaned\_text)

return summary

end

### **Check and Reward Goal**

| **Input: User ID, Goal ID** |
| --- |
| **Output:** Reward given (Boolean) |

function CheckAndRewardGoal(user\_id, goal\_id)

goal ← GetGoalDetails(goal\_id)

progress ← GetUserProgress(user\_id, goal.book, goal.chapter)

if progress ≥ goal.target then

reward ← AssignReward(user\_id)

return true

else

return false

end

pass

## **External APIs/SDKs**

Describe the third-party APIs/SDKs used in the project implementation in the following table. Few examples of APIs are provided in the table.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of API/SDK or Library** | **Description** | **Purpose of Usage** | **API Endpoint/Function Used** |
| **Cloudinary** | Image and video management solution | Uploading user-generated images (e.g., avatars, notes) to cloud storage | https://api.cloudinary.com/v1\_1/demo/image/upload |
| **Firebase Cloud Messaging (FCM)** | Messaging and push notification service | Sending notifications to users (e.g., quiz results, live class reminders) | FirebaseMessaging.onMessage, FirebaseMessaging.send() |
| **MongoDB Atlas Search** | Full-text search engine integrated with MongoDB | Searching books, topics, and flashcards across user data | db.collection.aggregate([{ $search: {...} }]) |
| **scikit-learn (Decision Tree)** | ML library for classification and regression | Classifying student performance and guiding personalized learning paths | DecisionTreeClassifier().fit(), .predict() |
| **scikit-learn (TF-IDF)** | Term Frequency–Inverse Document Frequency model | Identifying important keywords for AI flashcard and summary generation | TfidfVectorizer().fit\_transform() |
| **pyirt / IRT Model** | Item Response Theory model | Adaptive quiz difficulty and estimating student ability levels | irt\_model.fit(), irt\_model.predict() |
| **spaCy / Hugging Face Transformers** | Natural Language Processing (NLP) libraries | Summarizing educational content, generating explanations and paraphrased notes | nlp\_model(text), transformers.pipeline("summarization") |
| **Reinforcement Learning (Q-learning / Custom)** | Reward-based ML model | Used in dynamic battle matchmaking and progress optimization | env.step(), agent.learn() |

# **Testing and Evaluation**

## **Unit testing:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Description** | **Expected Result** | **Actual Result** | **pass /fail** |
| **UT\_BE\_Auth\_001** | Email format validation (valid) | Function returns `true`. | Function returns `true`. | pass |
| **UT\_BE\_Auth\_002** | Email format validation (invalid - no @) | Function returns `false`. | Function returns `false`. | pass |
| **UT\_BE\_Auth\_003** | Email format validation (invalid - no domain) | Function returns `false`. | Function returns `false`. | pass |
| **UT\_BE\_Auth\_004** | Email format validation (invalid - empty) | Function returns `false`. | Function returns `false`. | pass |
| **UT\_BE\_Auth\_005** | Email format validation (invalid - invalid characters) | Function returns `false`. | Function returns `false`. | pass |
| **UT\_BE\_Auth\_006** | Email format validation (invalid - multiple @) | Function returns `false`. | Function returns `false`. | pass |
| **UT\_BE\_Auth\_007** | Password hashing produces different output for same input (Salted Hash) | Functions return different hash strings (implies salting). | Functions return different hash strings (implies salting). | pass |
| **UT\_BE\_Auth\_008** | Password hashing produces non-empty output for valid input | Function returns a non-empty string. | Function returns a non-empty string. | pass |
| **UT\_BE\_Auth\_009** | Password comparison (correct password) | Function returns `true`. | Function returns `true`. | pass |
| **UT\_BE\_Auth\_010** | Password comparison (incorrect password) | Function returns `false`. | Function returns `false`. | pass |
| **UT\_BE\_Auth\_011** | Password comparison (empty password attempt) | Function returns `false`. | Function returns `false`. | pass |
| **UT\_BE\_Auth\_012** | User existence check (user exists by email - mocking DB) | Function returns `true`. | Function returns `true`. | pass |
| **UT\_BE\_Auth\_013** | User existence check (user exists by username - mocking DB) | Function returns `true`. | Function returns `true`. | pass |
| **UT\_BE\_Auth\_014** | User existence check (user does not exist - mocking DB) | Function returns `false`. | Function returns `false`. | pass |
| **UT\_BE\_Auth\_015** | Create user (successful - mocking DB insert) | Function successfully "inserts" user (in mock) and returns success status/ID. | Function successfully "inserts" user (in mock) and returns success status/ID. | pass |
| **UT\_BE\_Auth\_016** | Create user (failure - duplicate email from DB - mocking DB) | Function returns an error indicating duplicate email. | Function returns an error indicating duplicate email. | pass |
| **UT\_BE\_Auth\_017** | Create user (failure - duplicate username from DB - mocking DB) | Function returns an error indicating duplicate username. | Function returns an error indicating duplicate username. | pass |
| **UT\_BE\_Auth\_018** | Create user (failure - invalid email format - before DB) | Function returns an error indicating invalid email format. | Function returns an error indicating invalid email format. | pass |
| **UT\_BE\_Auth\_019** | Create user (failure - invalid password format/strength - before DB) | Function returns an error indicating invalid password. | Function returns an error indicating invalid password. | pass |
| **UT\_BE\_Auth\_020** | Find user by credentials (successful by email - mocking DB) | Function returns user data (excluding sensitive info). | Function returns user data (excluding sensitive info). | pass |
| **UT\_BE\_Auth\_021** | Find user by credentials (successful by username - mocking DB) | Function returns user data (excluding sensitive info). | Function returns user data (excluding sensitive info). | pass |
| **UT\_BE\_Auth\_022** | Find user by credentials (failure - incorrect password - mocking DB) | Function returns null or error indicating invalid credentials. | Function returns null or error indicating invalid credentials. | pass |
| **UT\_BE\_Auth\_023** | Find user by credentials (failure - user not found - mocking DB) | Function returns null or error indicating user not found. | Function returns null or error indicating user not found. | pass |
| **UT\_BE\_Auth\_024** | Minimum password length validation (valid) | Logic returns `true`. | Logic returns `true`. | pass |
| **UT\_BE\_Auth\_025** | Minimum password length validation (invalid) | Logic returns `false`. | Logic returns `false`. | pass |
| **UT\_BE\_Auth\_026** | Username character validation (valid) | Logic returns `true`. | Logic returns `true`. | pass |
| **UT\_BE\_Auth\_027** | Username character validation (invalid) | Logic returns `false`. | Logic returns `false`. | pass |
| **UT\_FE\_Auth\_028** | Input Field Component rendering | Component displays an input field with the correct placeholder. | Component displays an input field with the correct placeholder. | pass |
| **UT\_FE\_Auth\_029** | Password Input Component rendering with toggle | Component renders correctly and toggle icon changes visibility state as expected. | Component renders correctly and toggle icon changes visibility state as expected. | pass |
| **UT\_FE\_Auth\_030** | Button Component rendering (enabled state) | Component renders with the text "Sign In" and is visually active/clickable. | Component renders with the text "Sign In" and is visually active/clickable. | pass |
| **UT\_FE\_Auth\_031** | Button Component rendering (disabled state) | Component renders with the text "Sign In" and is visually greyed out/disabled. | Component renders with the text "Sign In" and is visually greyed out/disabled. | pass |
| **UT\_FE\_Auth\_032** | Client-side email validation logic (valid) | Function returns `true`. | Function returns `true`. | pass |
| **UT\_FE\_Auth\_033** | Client-side email validation logic (invalid) | Function returns `false`. | Function returns `false`. | pass |
| **UT\_FE\_Auth\_034** | Client-side password length validation logic (valid) | Logic returns `true`. | Logic returns `true`. | pass |
| **UT\_FE\_Auth\_035** | Client-side password length validation logic (invalid) | Logic returns `false`. | Logic returns `false`. | pass |
| **UT\_FE\_Onboard\_036** | Onboarding Page Component rendering | Component displays the provided dummy content correctly. | Component displays the provided dummy content correctly. | pass |
| **UT\_FE\_Onboard\_037** | Dot Indicator Component rendering (active state) | Component renders with the visual style indicating it's active. | Component renders with the visual style indicating it's active. | pass |
| **UT\_FE\_Onboard\_038** | Dot Indicator Component rendering (inactive state) | Component renders with the visual style indicating it's inactive. | Component renders with the visual style indicating it's inactive. | pass |
| **UT\_FE\_Home\_039** | Header Component rendering with dummy user data and icons | Component displays the username and icons correctly. | Component displays the username and icons correctly. | pass |
| **UT\_FE\_Home\_040** | Score Widget Component rendering with dummy data | Component displays the dummy score value and icon correctly. | Component displays the dummy score value and icon correctly. | pass |
| **UT\_FE\_Home\_041** | Subject Card Component rendering with dummy data | Component displays all provided dummy text, progress, and button correctly. | Component displays all provided dummy text, progress, and button correctly. | pass |
| **UT\_FE\_Home\_042** | Bottom Navigation Bar Component rendering (active state) | Component displays all navigation icons/labels, with "Home" visually active. | Component displays all navigation icons/labels, with "Home" visually active. | pass |
| **UT\_FE\_Home\_043** | Bottom Navigation Bar Component rendering (inactive state) | Component displays all navigation icons/labels, with "Home" visually inactive. | Component displays all navigation icons/labels, with "Home" visually inactive. | pass |
| **UT\_FE\_Modal\_044** | Generic Modal Component rendering with content and close button | Component renders a background overlay and a modal container with the title, content, and 'X' close button visible. | Component renders a background overlay and a modal container with the title, content, and 'X' close button visible. | pass |
| **UT\_FE\_Battle\_045** | Battle History Item Component rendering with dummy data | Component displays the initial, username, and arrow correctly. | Component displays the initial, username, and arrow correctly. | pass |
| **UT\_FE\_SelectSubject\_046** | Subject List Item Component rendering with dummy data (for Battle/Learn) | Component displays the subject name and icon correctly. | Component displays the subject name and icon correctly. | pass |
| **UT\_FE\_SelectSubject\_047** | Subject List Item Component rendering with dummy data (for Flashcards) | Component displays the subject name and flashcard count correctly. | Component displays the subject name and flashcard count correctly. | pass |
| **UT\_FE\_SelectChapter\_048** | Chapter Item Component rendering with dummy data and checkbox state | Component displays the chapter name and the checkbox in the correct checked/unchecked state. | Component displays the chapter name and the checkbox in the correct checked/unchecked state. | pass |
| **UT\_FE\_Battle\_049** | Brain Battle Question Component rendering | Component displays the question text correctly. | Component displays the question text correctly. | pass |
| **UT\_FE\_Battle\_050** | Brain Battle Answer Option Component rendering with state | Component displays label and text, applies correct visual styling based on `isSelected`, `isCorrect`, `isIncorrect` props (e.g., highlight, green/red background). | Component displays label and text, applies correct visual styling based on `isSelected`, `isCorrect`, `isIncorrect` props (e.g., highlight, green/red background). | pass |
| **UT\_FE\_ReportCard\_051** | Quiz Report Card Question Analysis Item Component rendering | Component displays all provided dummy data and the correct visual indicator (X for incorrect). | Component displays all provided dummy data and the correct visual indicator (X for incorrect). | pass |
| **UT\_FE\_ReportCard\_052** | Performance Metric Bar Component rendering | Component displays the label, percentage, and change indicator correctly, with a bar proportional to the percentage. | Component displays the label, percentage, and change indicator correctly, with a bar proportional to the percentage. | pass |
| **UT\_FE\_Classes\_053** | Class Card Component rendering with dummy data | Component displays all provided dummy data correctly. | Component displays all provided dummy data correctly. | pass |
| **UT\_FE\_ClassDetails\_054** | Class Details Action Button Component rendering | Component displays the icon, text, and forward arrow correctly and is clickable. | Component displays the icon, text, and forward arrow correctly and is clickable. | pass |
| **UT\_FE\_OverallScore\_055** | Overall Score Summary Widget rendering | Component displays all provided dummy data correctly with icons. | Component displays all provided dummy data correctly with icons. | pass |
| **UT\_FE\_OverallScore\_056** | Subject Performance Breakdown Item rendering | Component displays all provided dummy data correctly. | Component displays all provided dummy data correctly. | pass |
| **UT\_FE\_OverallScore\_057** | Area to Improve Item rendering with suggestions and button | Component displays the area, percentage, suggestions list, and button correctly. | Component displays the area, percentage, suggestions list, and button correctly. | pass |
| **UT\_FE\_SubjectPerf\_058** | Subject Progress Metric Item rendering | Component displays the value, label, and icon correctly. | Component displays the value, label, and icon correctly. | pass |
| **UT\_FE\_SubjectPerf\_059** | Chapter Performance Item rendering (collapsed) | Component displays the chapter name, status, percentage, and down arrow. | Component displays the chapter name, status, percentage, and down arrow. | pass |
| **UT\_FE\_SubjectPerf\_060** | Chapter Performance Item rendering (expanded with sub-topics) | Component displays the chapter name, status, percentage, up arrow, and renders the list of sub-topic items below it. | Component displays the chapter name, status, percentage, up arrow, and renders the list of sub-topic items below it. | pass |
| **UT\_FE\_SubjectPerf\_061** | Chapter Sub-topic Item rendering | Component displays the sub-topic name and percentage correctly with a checkmark icon. | Component displays the sub-topic name and percentage correctly with a checkmark icon. | pass |
| **UT\_FE\_Leaderboard\_062** | Leaderboard Podium Item rendering | Component displays the rank, user visual, name, and points correctly on the podium structure. | Component displays the rank, user visual, name, and points correctly on the podium structure. | pass |
| **UT\_FE\_Leaderboard\_063** | Leaderboard List Item Component rendering | Component displays the rank, user initial, name, points, and trophy icon correctly. | Component displays the rank, user initial, name, points, and trophy icon correctly. | pass |
| **UT\_FE\_Menu\_064** | Menu Item Component rendering with dummy data | Component displays the icon, title, and forward arrow correctly. | Component displays the icon, title, and forward arrow correctly. | pass |
| **UT\_FE\_Menu\_065** | Menu Item Component rendering (Logout) | Component displays the icon, title, and arrow/exit icon correctly. | Component displays the icon, title, and arrow/exit icon correctly. | pass |
| **UT\_FE\_Notifications\_066** | Notification Item Component rendering with dummy data | Component displays the icon, title, description, and timestamp correctly. | Component displays the icon, title, description, and timestamp correctly. | pass |

## **Functional testing:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | Test Description | Steps | Expected Result | Actual Result | Pass/Fail |
| **FT\_Onboarding\_001** | Verify initial Onboarding screen display | Launch the application after a fresh install (or simulating first launch). | The first onboarding screen (Image 1 content) is displayed prominently. Dot indicators show 1/3 active. | The first onboarding screen (Image 1 content) is displayed prominently. Dot indicators show 1/3 active. | pass |
| **FT\_Onboarding\_002** | Swipe left on 1st Onboarding screen | On the first onboarding screen, swipe left. | User is navigated to the second onboarding screen (Image 2 content). Dot indicators update to 2/3 active. | User is navigated to the second onboarding screen (Image 2 content). Dot indicators update to 2/3 active. | pass |
| **FT\_Onboarding\_003** | Swipe left on 2nd Onboarding screen | On the second onboarding screen, swipe left. | User is navigated to the third (last) onboarding screen (Image 3 content). Dot indicators update to 3/3 active. | User is navigated to the third (last) onboarding screen (Image 3 content). Dot indicators update to 3/3 active. | pass |
| **FT\_Onboarding\_004** | Swipe right on 2nd Onboarding screen | On the second onboarding screen, swipe right. | User is navigated back to the first onboarding screen (Image 1 content). Dot indicators update to 1/3 active. | User is navigated back to the first onboarding screen (Image 1 content). Dot indicators update to 1/3 active. | pass |
| **FT\_Onboarding\_005** | Swipe right on 3rd Onboarding screen | On the third onboarding screen, swipe right. | User is navigated back to the second onboarding screen (Image 2 content). Dot indicators update to 2/3 active. | User is navigated back to the second onboarding screen (Image 2 content). Dot indicators update to 2/3 active. | pass |
| **FT\_Onboarding\_006** | Attempt to swipe left on 3rd Onboarding screen | On the third onboarding screen, swipe left. | No navigation occurs. The user remains on the third screen. | No navigation occurs. The user remains on the third screen. | pass |
| **FT\_Onboarding\_007** | Tap a dot indicator (if interactive) | On any onboarding screen, tap one of the inactive dot indicators. | The user is directly navigated to the screen corresponding to the tapped dot. (Assumes dots are interactive, common UI pattern). | The user is directly navigated to the screen corresponding to the tapped dot. (Assumes dots are interactive, common UI pattern). | pass |
| **FT\_Login\_008** | Enter valid Username/Email in field | On Login screen, tap the "Username / Email" field and type "test@example.com". | The text "test@example.com" appears in the input field. Keyboard appears. | The text "test@example.com" appears in the input field. Keyboard appears. | pass |
| **FT\_Login\_009** | Enter valid Password in field | On Login screen, tap the "Password" field and type "password123". | The text "password123" appears in the input field, obscured by dots/asterisks. Keyboard appears. | The text "password123" appears in the input field, obscured by dots/asterisks. Keyboard appears. | pass |
| **FT\_Login\_010** | Toggle password visibility (if eye icon exists, not shown but common) | On Login screen, tap the password field toggle icon (if present). | The password text becomes visible (or hidden) as the icon changes state. | The password text becomes visible (or hidden) as the icon changes state. | pass |
| **FT\_Login\_011** | Click "Sign In" with valid credentials | On Login screen, enter valid credentials. Click "Sign In". | User is successfully logged in and navigated to the Home screen (Image 4). | User is successfully logged in and navigated to the Home screen (Image 4). | pass |
| **FT\_Login\_012** | Click "Sign In" with invalid password | On Login screen, enter valid Username/Email and invalid Password. Click "Sign In". | An error message (e.g., "Invalid credentials") is displayed on the screen. User remains on the Login screen. | An error message (e.g., "Invalid credentials") is displayed on the screen. User remains on the Login screen. | pass |
| **FT\_Login\_013** | Click "Sign In" with non-existent user | On Login screen, enter credentials for a non-existent user. Click "Sign In". | An error message (e.g., "User not found", "Invalid credentials") is displayed. User remains on the Login screen. | An error message (e.g., "User not found", "Invalid credentials") is displayed. User remains on the Login screen. | pass |
| **FT\_Login\_014** | Click "Sign In" with empty Username/Email | On Login screen, leave Username/Email empty, enter password. Click "Sign In". | Validation error message appears next to the empty field. Button click might be prevented if client-side validation active. User remains on screen. | Validation error message appears next to the empty field. Button click might be prevented if client-side validation active. User remains on screen. | pass |
| **FT\_Login\_015** | Click "Sign In" with empty Password | On Login screen, enter Username/Email, leave Password empty. Click "Sign In". | Validation error message appears next to the empty field. Button click might be prevented. User remains on screen. | Validation error message appears next to the empty field. Button click might be prevented. User remains on screen. | pass |
| **FT\_Login\_016** | Click "Sign In" with both fields empty | On Login screen, leave both fields empty. Click "Sign In". | Validation error messages appear for both fields. Button click might be prevented. User remains on screen. | Validation error messages appear for both fields. Button click might be prevented. User remains on screen. | pass |
| **FT\_Login\_017** | Click "Forgot Password?" link | On Login screen, click the "Forgot Password?" link. | User is navigated to a password recovery screen/flow. | User is navigated to a password recovery screen/flow. | pass |
| **FT\_Login\_018** | Click "Sign Up" link | On Login screen, click the "Sign Up" link. | User is navigated to the Sign Up screen (Image 3). | User is navigated to the Sign Up screen (Image 3). | pass |
| **FT\_Login\_019** | Click "Login with Apple" button | On Login screen, click "Login with Apple". | A system-level Apple login dialog is presented, or a placeholder action occurs. | A system-level Apple login dialog is presented, or a placeholder action occurs. | pass |
| **FT\_Login\_020** | Click "Login with Google" button | On Login screen, click "Login with Google". | A system-level Google login dialog is presented, or a placeholder action occurs. | A system-level Google login dialog is presented, or a placeholder action occurs. | pass |
| **FT\_Login\_021** | Click "Get Support" link | On Login screen, click the "Need help? Get Support" link. | User is navigated to a support screen or contact form (even if placeholder). | User is navigated to a support screen or contact form (even if placeholder). | pass |
| **FT\_Signup\_022** | Enter valid Full Name | On Sign Up screen, tap "Full Name" field and type "Test User". | Text appears correctly. Keyboard appears. | Text appears correctly. Keyboard appears. | pass |
| **FT\_Signup\_023** | Enter valid Username/Email | On Sign Up screen, tap "Username / Email" field and type "testuser@example.com". | Text appears correctly. Keyboard appears. | Text appears correctly. Keyboard appears. | pass |
| **FT\_Signup\_024** | Enter valid Password | On Sign Up screen, tap "Password" field and type "SecurePassword123". | Text appears correctly, obscured. Keyboard appears. | Text appears correctly, obscured. Keyboard appears. | pass |
| **FT\_Signup\_025** | Toggle password visibility (if eye icon exists) | On Sign Up screen, tap password field toggle icon (if present). | Password text becomes visible/hidden. Icon changes state. | Password text becomes visible/hidden. Icon changes state. | pass |
| **FT\_Signup\_026** | Click "Signup" with valid and unique data | On Sign Up screen, fill all fields with valid, unique data. Click "Signup". | User is successfully registered and navigated to Home screen or Login screen (as per design). | User is successfully registered and navigated to Home screen or Login screen (as per design). | pass |
| **FT\_Signup\_027** | Click "Signup" with existing email | On Sign Up screen, fill fields with valid data but an existing email. Click "Signup". | Error message (e.g., "Email already exists") is displayed. User remains on screen. | Error message (e.g., "Email already exists") is displayed. User remains on screen. | pass |
| **FT\_Signup\_028** | Click "Signup" with existing username | On Sign Up screen, fill fields with valid data but an existing username. Click "Signup". | Error message (e.g., "Username already exists") is displayed. User remains on screen. | Error message (e.g., "Username already exists") is displayed. User remains on screen. | pass |
| **FT\_Signup\_029** | Click "Signup" with invalid email format | On Sign Up screen, fill fields, enter invalid email format. Click "Signup". | Validation error message appears. User remains on screen. (Client-side or Backend validation feedback). | Validation error message appears. User remains on screen. (Client-side or Backend validation feedback). | pass |
| **FT\_Signup\_030** | Click "Signup" with password failing strength/length rules | On Sign Up screen, fill fields, enter a password failing rules. Click "Signup". | Validation error message appears. User remains on screen. (Client-side or Backend validation feedback). | Validation error message appears. User remains on screen. (Client-side or Backend validation feedback). | pass |
| **FT\_Signup\_031** | Click "Signup" with empty Full Name | On Sign Up screen, leave Full Name empty. Fill other fields. Click "Signup". | Validation error for Full Name appears. User remains on screen. | Validation error for Full Name appears. User remains on screen. | pass |
| **FT\_Signup\_032** | Click "Signup" with empty Username/Email | On Sign Up screen, leave Username/Email empty. Fill other fields. Click "Signup". | Validation error for Username/Email appears. User remains on screen. | Validation error for Username/Email appears. User remains on screen. | pass |
| **FT\_Signup\_033** | Click "Signup" with empty Password | On Sign Up screen, leave Password empty. Fill other fields. Click "Signup". | Validation error for Password appears. User remains on screen. | Validation error for Password appears. User remains on screen. | pass |
| **FT\_Signup\_034** | Click "Signup" with all fields empty | On Sign Up screen, leave all fields empty. Click "Signup". | Validation error for all required fields appears. User remains on screen. | Validation error for all required fields appears. User remains on screen. | pass |
| **FT\_Signup\_035** | Click "Log In" link | On Sign Up screen, click the "Log In" link. | User is navigated to the Login screen (Image 1 or 2). | User is navigated to the Login screen (Image 1 or 2). | pass |
| **FT\_Signup\_036** | Click "Login with Apple" button | On Sign Up screen, click "Login with Apple". | A system-level Apple login dialog is presented, or a placeholder action occurs. | A system-level Apple login dialog is presented, or a placeholder action occurs. | pass |
| **FT\_Signup\_037** | Click "Login with Google" button | On Sign Up screen, click "Login with Google". | A system-level Google login dialog is presented, or a placeholder action occurs. | A system-level Google login dialog is presented, or a placeholder action occurs. | pass |
| **FT\_Signup\_038** | Click "Get Support" link | On Sign Up screen, click the "Need help? Get Support" link. | User is navigated to a support screen or contact form (even if placeholder). | User is navigated to a support screen or contact form (even if placeholder). | pass |
| **\*\*Home Screen\*\*** |  |  |  |  | pass |
| **FT\_Home\_039** | Verify Welcome greeting and Username display | On Home screen, observe the greeting and username in the header. | The greeting (e.g., "Good Evening") is displayed. The logged-in user's name ("Nasir Bhutta") is displayed below it. | The greeting (e.g., "Good Evening") is displayed. The logged-in user's name ("Nasir Bhutta") is displayed below it. | pass |
| **FT\_Home\_040** | Verify Header Icons presence | On Home screen, observe the top right of the header. | A notification bell icon and a user initial/profile icon ('N') are displayed. | A notification bell icon and a user initial/profile icon ('N') are displayed. | pass |
| **FT\_Home\_041** | Verify Score widget display | On Home screen, observe the Score widget. | The widget displays an icon (trophy), a number (10), and the text "score". | The widget displays an icon (trophy), a number (10), and the text "score". | pass |
| **FT\_Home\_042** | Verify Coins widget display | On Home screen, observe the Coins widget. | The widget displays an icon (star), a number (1), and the icon text. | The widget displays an icon (star), a number (1), and the icon text. | pass |
| **FT\_Home\_043** | Verify Streak widget display | On Home screen, observe the Streak widget. | The widget displays an icon (flame), a number (0), and the icon text. | The widget displays an icon (flame), a number (0), and the icon text. | pass |
| **FT\_Home\_044** | Verify Lives widget display | On Home screen, observe the Lives widget. | The widget displays an icon (heart), a number (5/5), and the icon text. | The widget displays an icon (heart), a number (5/5), and the icon text. | pass |
| **FT\_Home\_045** | Click Score widget | On Home screen, click the Score widget (trophy). | The Score & Goals modal (Image 5) is displayed on top of the Home screen. | The Score & Goals modal (Image 5) is displayed on top of the Home screen. | pass |
| **FT\_Home\_046** | Click Coins widget | On Home screen, click the Coins widget (star). | The Coins modal (Image 7) is displayed. | The Coins modal (Image 7) is displayed. | pass |
| **FT\_Home\_047** | Click Streak widget | On Home screen, click the Streak widget (flame). | The Streak modal (Image 8) is displayed. | The Streak modal (Image 8) is displayed. | pass |
| **FT\_Home\_048** | Click Lives widget | On Home screen, click the Lives widget (heart). | The Lives modal (Image 9) is displayed. | The Lives modal (Image 9) is displayed. | pass |
| **FT\_Home\_049** | Verify Bookmark card display | On Home screen, observe the Bookmark card. | A card titled "Bookmark 6 Questions" with a "Claim Now" button is displayed. An illustration is present. | A card titled "Bookmark 6 Questions" with a "Claim Now" button is displayed. An illustration is present. | pass |
| **FT\_Home\_050** | Click "Claim Now" on Bookmark card | On Home screen, click the "Claim Now" button on the Bookmark card. | A placeholder action occurs (e.g., a toast message "Claimed!" or navigation to a bookmarks screen). The button might become disabled or change text. | A placeholder action occurs (e.g., a toast message "Claimed!" or navigation to a bookmarks screen). The button might become disabled or change text. | pass |
| **FT\_Home\_051** | Verify Subject cards display | On Home screen, scroll down and observe the subject cards. | Subject cards for Mathematics, Physics, and Biology (and potentially others if scrollable) are displayed, showing subject name, level, description, progress bar, and "Answer Quiz" button. Illustrations are present. | Subject cards for Mathematics, Physics, and Biology (and potentially others if scrollable) are displayed, showing subject name, level, description, progress bar, and "Answer Quiz" button. Illustrations are present. | pass |
| **FT\_Home\_052** | Click "Answer Quiz" on Mathematics card | On Home screen, click the "Answer Quiz" button on the Mathematics card. | User is navigated to a quiz screen related to Mathematics (e.g., Brain Battle screen or a specific quiz entry point). Context (subject: Mathematics) should be passed. | User is navigated to a quiz screen related to Mathematics (e.g., Brain Battle screen or a specific quiz entry point). Context (subject: Mathematics) should be passed. | pass |
| **FT\_Home\_053** | Click "Answer Quiz" on Physics card | On Home screen, click the "Answer Quiz" button on the Physics card. | User is navigated to a quiz screen related to Physics. Context (subject: Physics) should be passed. | User is navigated to a quiz screen related to Physics. Context (subject: Physics) should be passed. | pass |
| **FT\_Home\_054** | Click "Answer Quiz" on Biology card | On Home screen, click the "Answer Quiz" button on the Biology card. | User is navigated to a quiz screen related to Biology. Context (subject: Biology) should be passed. | User is navigated to a quiz screen related to Biology. Context (subject: Biology) should be passed. | pass |
| **FT\_Home\_055** | Click notification icon in header | On Home screen, click the notification bell icon. | User is navigated to the Notifications screen (Image 31). | User is navigated to the Notifications screen (Image 31). | pass |
| **FT\_Home\_056** | Click user initial/profile icon in header | On Home screen, click the 'N' icon. | User is navigated to the Menu screen (Image 30). | User is navigated to the Menu screen (Image 30). | pass |
| **\*\*Bottom Navigation Bar\*\*** |  |  |  |  | pass |
| **FT\_BottomNav\_057** | Click "Home" when on Home screen | On Home screen, tap "Home" in the bottom nav. | No navigation occurs. The Home screen remains active. (Unless designed to refresh). | No navigation occurs. The Home screen remains active. (Unless designed to refresh). | pass |
| **FT\_BottomNav\_058** | Click "Learn" from Home screen | On Home screen, tap "Learn" in the bottom nav. | The "Learn" icon becomes active. User is navigated to a Learn-related screen (likely subject selection, similar to Image 12 or 22). | The "Learn" icon becomes active. User is navigated to a Learn-related screen (likely subject selection, similar to Image 12 or 22). | pass |
| **FT\_BottomNav\_059** | Click "Class" from Home screen | On Home screen, tap "Class" in the bottom nav. | The "Class" icon becomes active. User is navigated to the My Classes screen (Image 23). | The "Class" icon becomes active. User is navigated to the My Classes screen (Image 23). | pass |
| **FT\_BottomNav\_060** | Click "Extras" from Home screen | On Home screen, tap "Extras" in the bottom nav. | The "Extras" icon becomes active. The Extras menu modal (Image 10) is displayed on top of the Home screen. | The "Extras" icon becomes active. The Extras menu modal (Image 10) is displayed on top of the Home screen. | pass |
| **FT\_BottomNav\_061** | Click "Home" from Class screen | Navigate to Class screen (FT\_BottomNav\_059). Tap "Home" in the bottom nav. | The "Home" icon becomes active. User is navigated back to the Home screen (Image 4). | The "Home" icon becomes active. User is navigated back to the Home screen (Image 4). | pass |
| **FT\_BottomNav\_062** | Click "Learn" from Class screen | Navigate to Class screen. Tap "Learn" in the bottom nav. | The "Learn" icon becomes active. User is navigated to the Learn screen. | The "Learn" icon becomes active. User is navigated to the Learn screen. | pass |
| **FT\_BottomNav\_063** | Click "Extras" from Class screen | Navigate to Class screen. Tap "Extras" in the bottom nav. | The "Extras" icon becomes active. The Extras modal appears over the Class screen. | The "Extras" icon becomes active. The Extras modal appears over the Class screen. | pass |
| **\*\*Extras Modal\*\*** |  |  |  |  | pass |
| **FT\_Extras\_064** | Verify Extras modal content | Trigger Extras modal (FT\_BottomNav\_060/063). | The modal displays icons and text for "Battle", "Practice", "FlashCards", and "Books". An 'X' close button is in the corner. | The modal displays icons and text for "Battle", "Practice", "FlashCards", and "Books". An 'X' close button is in the corner. | pass |
| **FT\_Extras\_065** | Click "Battle" in Extras modal | Open Extras modal. Click "Battle". | Modal closes. User is navigated to the Battle Lobby screen (Image 11). | Modal closes. User is navigated to the Battle Lobby screen (Image 11). | pass |
| **FT\_Extras\_066** | Click "Practice" in Extras modal | Open Extras modal. Click "Practice". | Modal closes. User is navigated to a Practice subject selection screen (similar to Image 12). | Modal closes. User is navigated to a Practice subject selection screen (similar to Image 12). | pass |
| **FT\_Extras\_067** | Click "FlashCards" in Extras modal | Open Extras modal. Click "FlashCards". | Modal closes. User is navigated to the Flashcard subject selection screen (Image 22). | Modal closes. User is navigated to the Flashcard subject selection screen (Image 22). | pass |
| **FT\_Extras\_068** | Click "Books" in Extras modal | Open Extras modal. Click "Books". | Modal closes. User is navigated to a Books/Study Materials screen (even if dummy). | Modal closes. User is navigated to a Books/Study Materials screen (even if dummy). | pass |
| **FT\_Extras\_069** | Click 'X' to close Extras modal | Open Extras modal. Click the 'X' button. | Modal is dismissed. User returns to the screen from which the modal was opened. | Modal is dismissed. User returns to the screen from which the modal was opened. | pass |
| **FT\_Extras\_070** | Tap outside Extras modal to close (if applicable) | Open Extras modal. Tap on the dimmed background outside the modal area. | If implemented, modal is dismissed. User returns to the screen below. | If implemented, modal is dismissed. User returns to the screen below. | pass |
| **\*\*Battle Flow\*\*** |  |  |  |  | pass |
| **FT\_BattleLobby\_071** | Verify Battle Lobby screen layout and content | Navigate to Battle Lobby (FT\_Extras\_065). | Screen displays header with back button ("<") and title "Battle". User info area (initial, username, stats). Banner. "Start the Battle", "Enter Invitation Code" buttons. "Your Battle History" section title and dummy list. | Screen displays header with back button ("<") and title "Battle". User info area (initial, username, stats). Banner. "Start the Battle", "Enter Invitation Code" buttons. "Your Battle History" section title and dummy list. | pass |
| **FT\_BattleLobby\_072** | Click "Start the Battle" button | On Battle Lobby, click "Start the Battle". | User is navigated to the Select Subject screen (Image 12). | User is navigated to the Select Subject screen (Image 12). | pass |
| **FT\_BattleLobby\_073** | Click "Enter Invitation Code" button | On Battle Lobby, click "Enter Invitation Code". | The "Battle invitation" modal (Image 21) is displayed on top. | The "Battle invitation" modal (Image 21) is displayed on top. | pass |
| **FT\_BattleLobby\_074** | Click a Battle History list item | On Battle Lobby, click one of the dummy items in "Your Battle History". | User is navigated to a Battle Report Card or history details screen (Image 19/20 equivalent) related to that history item. | User is navigated to a Battle Report Card or history details screen (Image 19/20 equivalent) related to that history item. | pass |
| **FT\_BattleLobby\_075** | Click back button on Battle Lobby | On Battle Lobby, click the "<" back button. | User is navigated back to the previous screen (likely Home). | User is navigated back to the previous screen (likely Home). | pass |
| **FT\_BattleInvitation\_076** | Verify Battle Invitation modal content | Open Battle Invitation modal (FT\_BattleLobby\_073). | Modal displays title "Battle invitation", instruction text, input fields for code, and "Enter match" button. 'X' close button is present. | Modal displays title "Battle invitation", instruction text, input fields for code, and "Enter match" button. 'X' close button is present. | pass |
| **FT\_BattleInvitation\_077** | Enter code in Invitation modal fields | Open Invitation modal. Tap fields and type text. | Text appears in the fields. Keyboard appears. | Text appears in the fields. Keyboard appears. | pass |
| **FT\_BattleInvitation\_078** | Click "Enter match" in Invitation modal (dummy action) | Open Invitation modal. Enter text (valid/invalid format) in fields. Click "Enter match". | A placeholder action occurs (e.g., message "Attempting to join match with code..."). Navigation might be attempted to a dummy battle screen or error message displayed based on dummy logic. | A placeholder action occurs (e.g., message "Attempting to join match with code..."). Navigation might be attempted to a dummy battle screen or error message displayed based on dummy logic. | pass |
| **FT\_BattleInvitation\_079** | Click 'X' to close Battle Invitation modal | Open Invitation modal. Click the 'X' button. | Modal is dismissed. User returns to the Battle Lobby screen. | Modal is dismissed. User returns to the Battle Lobby screen. | pass |
| **FT\_SelectSubjectBattle\_080** | Verify Select Subject screen content (from Battle) | Navigate to Select Subject from Battle (FT\_BattleLobby\_072). | Screen displays header ("Select Subject") and list of subjects (English, Biology, etc.) with icons. | Screen displays header ("Select Subject") and list of subjects (English, Biology, etc.) with icons. | pass |
| **FT\_SelectSubjectBattle\_081** | Click a subject on Select Subject (from Battle) | On Select Subject screen, click "Mathematics". | The "Battle Mode" dialog (Image 13) is displayed, mentioning the selected subject "Mathematics". | The "Battle Mode" dialog (Image 13) is displayed, mentioning the selected subject "Mathematics". | pass |
| **FT\_SelectSubjectBattle\_082** | Click back button on Select Subject | On Select Subject, click "<". | User is navigated back to the Battle Lobby screen. | User is navigated back to the Battle Lobby screen. | pass |
| **FT\_BattleModeDialog\_083** | Verify Battle Mode dialog content | Open Battle Mode dialog (FT\_SelectSubjectBattle\_081). | Dialog displays title "Battle Mode", question "How would you like to compete...", and buttons "By Chapter", "Whole Book". 'X' close button is present. | Dialog displays title "Battle Mode", question "How would you like to compete...", and buttons "By Chapter", "Whole Book". 'X' close button is present. | pass |
| **FT\_BattleModeDialog\_084** | Click "By Chapter" in Battle Mode dialog | Open Battle Mode dialog. Click "By Chapter". | Modal closes. User is navigated to the "Select Chapters - Mathematics" screen (Image 15). | Modal closes. User is navigated to the "Select Chapters - Mathematics" screen (Image 15). | pass |
| **FT\_BattleModeDialog\_085** | Click "Whole Book" in Battle Mode dialog | Open Battle Mode dialog. Click "Whole Book". | Modal closes. The "Invite Friends" dialog (Image 14) is displayed. | Modal closes. The "Invite Friends" dialog (Image 14) is displayed. | pass |
| **FT\_BattleModeDialog\_086** | Click 'X' to close Battle Mode dialog | Open Battle Mode dialog. Click the 'X'. | Modal is dismissed. User returns to the Select Subject screen. | Modal is dismissed. User returns to the Select Subject screen. | pass |
| **FT\_InviteFriendsDialog\_087** | Verify Invite Friends dialog content | Open Invite Friends dialog (FT\_BattleModeDialog\_085). | Dialog displays title "Invite Friends", question "Are you ready?", and buttons "Share invitation code", "Random Match". 'X' close button is present. | Dialog displays title "Invite Friends", question "Are you ready?", and buttons "Share invitation code", "Random Match". 'X' close button is present. | pass |
| **FT\_InviteFriendsDialog\_088** | Click "Share invitation code" in dialog (dummy action) | Open Invite Friends dialog. Click "Share invitation code". | A system-level share sheet or placeholder appears (e.g., message "Sharing code..."). | A system-level share sheet or placeholder appears (e.g., message "Sharing code..."). | pass |
| **FT\_InviteFriendsDialog\_089** | Click "Random Match" in dialog (dummy action) | Open Invite Friends dialog. Click "Random Match". | Modal closes. User is navigated to the Brain Battle screen (Image 16), simulating a random match start. | Modal closes. User is navigated to the Brain Battle screen (Image 16), simulating a random match start. | pass |
| **FT\_InviteFriendsDialog\_090** | Click 'X' to close Invite Friends dialog | Open Invite Friends dialog. Click the 'X'. | Modal is dismissed. User returns to the Select Subject screen (where Battle Mode dialog originated). | Modal is dismissed. User returns to the Select Subject screen (where Battle Mode dialog originated). | pass |
| **FT\_SelectChapters\_091** | Verify Select Chapters screen layout and content | Navigate to Select Chapters (FT\_BattleModeDialog\_084). | Screen displays header ("Select Chapters - Mathematics") with back button. A list of chapters with checkboxes is shown (Algebra, Geometry, etc.). Below, "Selected Chapters: X" count. "Start Match" button below, initially disabled. | Screen displays header ("Select Chapters - Mathematics") with back button. A list of chapters with checkboxes is shown (Algebra, Geometry, etc.). Below, "Selected Chapters: X" count. "Start Match" button below, initially disabled. | pass |
| **FT\_SelectChapters\_092** | Click a chapter checkbox (select) | On Select Chapters, click the "Algebra" checkbox. | The checkbox becomes checked. The "Selected Chapters" count increases by 1. The "Start Match" button becomes enabled. | The checkbox becomes checked. The "Selected Chapters" count increases by 1. The "Start Match" button becomes enabled. | pass |
| **FT\_SelectChapters\_093** | Click a chapter checkbox (deselect) | On Select Chapters, click the "Algebra" checkbox which is currently checked. | The checkbox becomes unchecked. The "Selected Chapters" count decreases by 1. The "Start Match" button might become disabled if count is 0. | The checkbox becomes unchecked. The "Selected Chapters" count decreases by 1. The "Start Match" button might become disabled if count is 0. | pass |
| **FT\_SelectChapters\_094** | Select multiple chapters | On Select Chapters, select "Algebra", then "Geometry". | Both checkboxes are checked. "Selected Chapters" count is 2. "Start Match" button remains enabled. | Both checkboxes are checked. "Selected Chapters" count is 2. "Start Match" button remains enabled. | pass |
| **FT\_SelectChapters\_095** | Click "Start Match" with chapters selected | On Select Chapters, ensure at least one chapter is selected. Click "Start Match". | User is navigated to the Brain Battle screen (Image 16), simulating a match starting with questions from selected chapters. | User is navigated to the Brain Battle screen (Image 16), simulating a match starting with questions from selected chapters. | pass |
| **FT\_SelectChapters\_096** | Attempt to click "Start Match" with no chapters selected | On Select Chapters, ensure no chapters are selected. Attempt to click the "Start Match" button. | The button does not respond to clicks (it's disabled). | The button does not respond to clicks (it's disabled). | pass |
| **FT\_SelectChapters\_097** | Click back button on Select Chapters | On Select Chapters, click "<". | User is navigated back to the Select Subject screen. | User is navigated back to the Select Subject screen. | pass |
| **FT\_BrainBattle\_098** | Verify Brain Battle screen initial state | Navigate to Brain Battle (FT\_SelectChapters\_095 or FT\_InviteFriendsDialog\_089). | Screen displays header ("Brain Battle", back button), question number ("Question 1/5"), scores ("You 0 vs Opponent 0"), Time Remaining bar (full) and text (e.g., "15 seconds"), question text, and 4 answer options (A, B, C, D). | Screen displays header ("Brain Battle", back button), question number ("Question 1/5"), scores ("You 0 vs Opponent 0"), Time Remaining bar (full) and text (e.g., "15 seconds"), question text, and 4 answer options (A, B, C, D). | pass |
| **FT\_BrainBattle\_099** | Select an Answer Option | On Brain Battle, click one of the answer options (e.g., Option C). | The clicked option is highlighted. The screen progresses to the next question after a brief delay. Your score and opponent score may update (dummy). Question number increments. Timer resets. | The clicked option is highlighted. The screen progresses to the next question after a brief delay. Your score and opponent score may update (dummy). Question number increments. Timer resets. | pass |
| **FT\_BrainBattle\_100** | Observe Time Remaining decrement | On Brain Battle, watch the Time Remaining bar and text. | The bar visually decreases and the text counts down towards 0. | The bar visually decreases and the text counts down towards 0. | pass |
| **FT\_BrainBattle\_101** | Timer runs out on a question | On Brain Battle, wait for the timer to reach 0 without selecting an answer. | The UI indicates time is up (e.g., bar empty, text 0s). The screen progresses to the next question after a brief delay. Your score does not increase for this question. | The UI indicates time is up (e.g., bar empty, text 0s). The screen progresses to the next question after a brief delay. Your score does not increase for this question. | pass |
| **FT\_BrainBattle\_102** | Complete the Battle (answer all 5 questions) | Proceed through the Brain Battle, answering all 5 questions or letting the timer expire. | After the 5th question, the Brain Battle screen transitions to the Battle Result modal (Image 18). | After the 5th question, the Brain Battle screen transitions to the Battle Result modal (Image 18). | pass |
| **FT\_BrainBattle\_103** | Click back button during Battle | On Brain Battle screen, click the "<" back button. | A confirmation dialog (e.g., "Are you sure you want to leave? Your progress will be lost.") is displayed. (Common pattern for ongoing activities). | A confirmation dialog (e.g., "Are you sure you want to leave? Your progress will be lost.") is displayed. (Common pattern for ongoing activities). | pass |
| **FT\_BrainBattle\_104** | Confirm leaving Battle in dialog (if dialog appears) | If leaving dialog appears (FT\_BrainBattle\_103), click "Yes" or "Leave". | The user is navigated back to the previous screen (e.g., Battle Lobby). The battle state is abandoned. | The user is navigated back to the previous screen (e.g., Battle Lobby). The battle state is abandoned. | pass |
| **FT\_BrainBattle\_105** | Cancel leaving Battle in dialog (if dialog appears) | If leaving dialog appears (FT\_BrainBattle\_103), click "No" or "Cancel". | The dialog is dismissed. User remains on the Brain Battle screen. | The dialog is dismissed. User remains on the Brain Battle screen. | pass |
| **FT\_BattleResult\_106** | Verify Battle Result modal content | Complete a battle (FT\_BrainBattle\_102). | Modal displays title "Battle Result", a result trophy icon, a win/loss/draw status ("You Lost!"). It shows "Your Score" and "Opponent Score" with numerical values. A "Done" button is present. | Modal displays title "Battle Result", a result trophy icon, a win/loss/draw status ("You Lost!"). It shows "Your Score" and "Opponent Score" with numerical values. A "Done" button is present. | pass |
| **FT\_BattleResult\_107** | Click "Done" button in Battle Result modal | On Battle Result modal, click "Done". | Modal is dismissed. User is navigated to the Quiz Report Card screen (Image 19). | Modal is dismissed. User is navigated to the Quiz Report Card screen (Image 19). | pass |
| **FT\_ReportCard\_108** | Verify Report Card header and summary section | Navigate to Quiz Report Card (FT\_BattleResult\_107 or FT\_BattleLobby\_074). | Screen displays header ("Quiz Report Card", back button, download, share icons). A summary section shows result status (Victory/Loss), completion date/time, Your Score, Opponent Score, and Time Taken (dummy data). | Screen displays header ("Quiz Report Card", back button, download, share icons). A summary section shows result status (Victory/Loss), completion date/time, Your Score, Opponent Score, and Time Taken (dummy data). | pass |
| **FT\_ReportCard\_109** | Verify Performance Summary section | On Report Card, observe Performance Summary section. | Section displays "Accuracy" (%), "Correct" (count), and "Avg Time" (seconds) (dummy data). | Section displays "Accuracy" (%), "Correct" (count), and "Avg Time" (seconds) (dummy data). | pass |
| **FT\_ReportCard\_110** | Verify Question Analysis section layout | On Report Card, observe Question Analysis section. | Section is titled "Question Analysis" and shows a list of questions. Each item has a number, question text, "Your Answer", "Correct Answer", and difficulty. | Section is titled "Question Analysis" and shows a list of questions. Each item has a number, question text, "Your Answer", "Correct Answer", and difficulty. | pass |
| **FT\_ReportCard\_111** | Verify Question Analysis item (Incorrect answer) | On Report Card, find an item for an incorrectly answered question (e.g., Question 1 in Image 19). | The question text, "Your Answer" (e.g., "Berlin"), and "Correct Answer" (e.g., "Paris") are displayed. An 'X' icon or red indicator is shown next to the question. Difficulty is shown. | The question text, "Your Answer" (e.g., "Berlin"), and "Correct Answer" (e.g., "Paris") are displayed. An 'X' icon or red indicator is shown next to the question. Difficulty is shown. | pass |
| **FT\_ReportCard\_112** | Verify Question Analysis item (Correct answer - not shown but assumed) | On Report Card, find an item for a correctly answered question (if dummy data includes one). | The question text, "Your Answer", and "Correct Answer" are displayed (they should be the same). A checkmark icon or green indicator is shown. Difficulty is shown. | The question text, "Your Answer", and "Correct Answer" are displayed (they should be the same). A checkmark icon or green indicator is shown. Difficulty is shown. | pass |
| **FT\_ReportCard\_113** | Verify Performance Metrics section | On Report Card, scroll to Performance Metrics section (Image 20). | Section titled "Performance Metrics". Bar graphs for Accuracy, Speed, and Score are displayed with dummy percentages/values and change indicators. | Section titled "Performance Metrics". Bar graphs for Accuracy, Speed, and Score are displayed with dummy percentages/values and change indicators. | pass |
| **FT\_ReportCard\_114** | Verify Comparison with Past Quizzes section | On Report Card, scroll to Comparison with Past Quizzes section (Image 20). | Section titled "Comparison with Past Quizzes". A chart (bar graph) compares current vs past quizzes (dummy data). A table below shows quiz details (Date, Score, Accuracy, Time/Q) for current and past quizzes (dummy data). | Section titled "Comparison with Past Quizzes". A chart (bar graph) compares current vs past quizzes (dummy data). A table below shows quiz details (Date, Score, Accuracy, Time/Q) for current and past quizzes (dummy data). | pass |
| **FT\_ReportCard\_115** | Click back button on Report Card | On Report Card, click "<". | User is navigated back to the screen from which the report card was accessed (likely Battle Lobby or Home). | User is navigated back to the screen from which the report card was accessed (likely Battle Lobby or Home). | pass |
| **FT\_ReportCard\_116** | Click Download icon | On Report Card, click the download icon in the header. | A placeholder action related to downloading the report occurs (e.g., a toast, a modal, or initiating download - dummy logic). | A placeholder action related to downloading the report occurs (e.g., a toast, a modal, or initiating download - dummy logic). | pass |
| **FT\_ReportCard\_117** | Click Share icon | On Report Card, click the share icon in the header. | A system-level share sheet or placeholder appears to share the report (dummy logic). | A system-level share sheet or placeholder appears to share the report (dummy logic). | pass |
| **FT\_Learn\_118** | Navigate to Learn subject selection | Tap "Learn" in bottom nav (FT\_BottomNav\_058). | User is navigated to a Select Subject screen (similar to Image 12), titled "Select Subject", listing subjects with icons. | User is navigated to a Select Subject screen (similar to Image 12), titled "Select Subject", listing subjects with icons. | pass |
| **FT\_Learn\_119** | Click a subject on Learn selection screen | On Learn subject selection, click "Physics". | User is navigated to a Physics-specific learning screen (e.g., chapter list, learning materials, quiz list) (even if dummy). Context (subject: Physics) passed. | User is navigated to a Physics-specific learning screen (e.g., chapter list, learning materials, quiz list) (even if dummy). Context (subject: Physics) passed. | pass |
| **FT\_Learn\_120** | Click back button on Learn subject selection | On Learn subject selection, click "<". | User is navigated back to the Home screen. | User is navigated back to the Home screen. | pass |
| **FT\_Flashcards\_121** | Navigate to Flashcard subject selection | Tap "FlashCards" in Extras modal (FT\_Extras\_067). | User is navigated to the Select Subject screen for Flashcards (Image 22), titled "Select Subject". List of subjects with icons and flashcard counts (dummy). | User is navigated to the Select Subject screen for Flashcards (Image 22), titled "Select Subject". List of subjects with icons and flashcard counts (dummy). | pass |
| **FT\_Flashcards\_122** | Click a subject on Flashcard selection screen | On Flashcard selection (Image 22), click "English". | User is navigated to the Flashcards screen for English (even if dummy flashcards). Context (subject: English) passed. | User is navigated to the Flashcards screen for English (even if dummy flashcards). Context (subject: English) passed. | pass |
| **FT\_Flashcards\_123** | Click back button on Flashcard subject selection | On Flashcard selection (Image 22), click "<". | User is navigated back to the screen from which the Flashcards screen was opened (likely Home or Battle Lobby if accessed via Home->Extras->Flashcards). | User is navigated back to the screen from which the Flashcards screen was opened (likely Home or Battle Lobby if accessed via Home->Extras->Flashcards). | pass |
| **FT\_Classes\_124** | Navigate to My Classes screen | Tap "Class" in bottom nav (FT\_BottomNav\_059). | User is navigated to the My Classes screen (Image 23), titled "My Classes". "Enrolled Classes" title and a list of class cards are displayed. | User is navigated to the My Classes screen (Image 23), titled "My Classes". "Enrolled Classes" title and a list of class cards are displayed. | pass |
| **FT\_Classes\_125** | Verify My Classes list content | On My Classes screen, observe the list of class cards. | Cards for Mathematics, Science, and English Literature are shown, each with dummy teacher, schedule, and progress (e.g., 5/12). | Cards for Mathematics, Science, and English Literature are shown, each with dummy teacher, schedule, and progress (e.g., 5/12). | pass |
| **FT\_Classes\_126** | Click a class card | On My Classes screen, click the "Mathematics" class card. | User is navigated to the Class Details screen for Mathematics (Image 24). Context (class: Mathematics) passed. | User is navigated to the Class Details screen for Mathematics (Image 24). Context (class: Mathematics) passed. | pass |
| **FT\_Classes\_127** | Click back button on My Classes | On My Classes screen, click "<". | User is navigated back to the Home screen. | User is navigated back to the Home screen. | pass |
| **FT\_ClassDetails\_128** | Verify Class Details screen layout and content | Navigate to Class Details (FT\_Classes\_126). | Screen displays header ("Mathematics", back button), Teacher name ("Mr. Johnson"). "Details" section (Schedule, Students - dummy). "Progress" section (Assignments completion - dummy). "Actions" section with buttons "View Materials", "Assignments", "Forum", "Contact Teacher". | Screen displays header ("Mathematics", back button), Teacher name ("Mr. Johnson"). "Details" section (Schedule, Students - dummy). "Progress" section (Assignments completion - dummy). "Actions" section with buttons "View Materials", "Assignments", "Forum", "Contact Teacher". | pass |
| **FT\_ClassDetails\_129** | Click "View Materials" button | On Class Details, click "View Materials". | User is navigated to a Class Materials screen (even if dummy content). | User is navigated to a Class Materials screen (even if dummy content). | pass |
| **FT\_ClassDetails\_130** | Click "Assignments" button | On Class Details, click "Assignments". | User is navigated to a Class Assignments screen (even if dummy content). | User is navigated to a Class Assignments screen (even if dummy content). | pass |
| **FT\_ClassDetails\_131** | Click "Forum" button | On Class Details, click "Forum". | User is navigated to a Class Forum screen (even if dummy content). | User is navigated to a Class Forum screen (even if dummy content). | pass |
| **FT\_ClassDetails\_132** | Click "Contact Teacher" button | On Class Details, click "Contact Teacher". | A contact interface (e.g., email composer pre-filled, message form modal/screen - dummy) appears, targeting the teacher. | A contact interface (e.g., email composer pre-filled, message form modal/screen - dummy) appears, targeting the teacher. | pass |
| **FT\_ClassDetails\_133** | Click back button on Class Details | On Class Details, click "<". | User is navigated back to the My Classes screen. | User is navigated back to the My Classes screen. | pass |
| **FT\_OverallScore\_134** | Navigate to Overall Score screen | Navigate via Menu (FT\_Menu\_162). | User is navigated to Overall Score screen (Image 25), titled "Overall Score". Header back button present. Overall Score section, Subject Breakdown/Areas to Improve tabs (Subject Breakdown active), Subject Breakdown content (chart, list). | User is navigated to Overall Score screen (Image 25), titled "Overall Score". Header back button present. Overall Score section, Subject Breakdown/Areas to Improve tabs (Subject Breakdown active), Subject Breakdown content (chart, list). | pass |
| **FT\_OverallScore\_135** | Verify Overall Score section content | On Overall Score, observe top section. | Displays "Your Overall Score" (Grade, Percentage, Score Ring) and summary metrics (Books, Quizzes, Hours) with icons (dummy data). | Displays "Your Overall Score" (Grade, Percentage, Score Ring) and summary metrics (Books, Quizzes, Hours) with icons (dummy data). | pass |
| **FT\_OverallScore\_136** | Verify Subject Breakdown tab content | On Overall Score, ensure "Subject Breakdown" tab is active. | Displays performance bar chart by subject (Mathematics, English, History Geography - dummy data). List of subjects below with completion stats and grade (dummy data). | Displays performance bar chart by subject (Mathematics, English, History Geography - dummy data). List of subjects below with completion stats and grade (dummy data). | pass |
| **FT\_OverallScore\_137** | Switch to "Areas to Improve" tab | On Overall Score, click the "Areas to Improve" tab. | The "Areas to Improve" tab becomes active. The content below the tabs updates to display areas needing improvement (Image 26). | The "Areas to Improve" tab becomes active. The content below the tabs updates to display areas needing improvement (Image 26). | pass |
| **FT\_OverallScore\_138** | Verify "Areas to Improve" tab content | On Overall Score, ensure "Areas to Improve" tab is active. | Section titled "Areas that need improvement". Lists specific areas (Geography - Map Reading, Mathematics - Algebra) with percentages (dummy), improvement suggestions, and "Practice Now" buttons. | Section titled "Areas that need improvement". Lists specific areas (Geography - Map Reading, Mathematics - Algebra) with percentages (dummy), improvement suggestions, and "Practice Now" buttons. | pass |
| **FT\_OverallScore\_139** | Click "Practice Now" button on Areas to Improve | On Overall Score ("Areas to Improve" tab), click "Practice Now" for "Geography - Map Reading". | User is navigated to a practice screen/quiz specifically for Geography Map Reading (even if dummy). Context passed. | User is navigated to a practice screen/quiz specifically for Geography Map Reading (even if dummy). Context passed. | pass |
| **FT\_OverallScore\_140** | Click a subject item in "Subject Breakdown" (e.g., Mathematics) | On Overall Score ("Subject Breakdown" tab), click the "Mathematics" list item. | User is navigated to the Subject Performance screen for Mathematics (Image 27). Context passed. | User is navigated to the Subject Performance screen for Mathematics (Image 27). Context passed. | pass |
| **FT\_OverallScore\_141** | Click back button on Overall Score | On Overall Score, click "<". | User is navigated back to the screen from which the Overall Score was accessed (likely Menu). | User is navigated back to the screen from which the Overall Score was accessed (likely Menu). | pass |
| **\*\*Subject Performance Screen\*\*** |  |  |  |  | pass |
| **FT\_SubjectPerf\_142** | Verify Subject Performance screen layout and content | Navigate to Subject Performance (FT\_OverallScore\_140 or FT\_Home\_052 etc. if subjects on Home navigate here). | Screen displays header (Subject Title, back button). Course card (title, instructor, grade, percentage - dummy). Progress metrics (Pages Read, Study Time, Quizzes - dummy). Score Breakdown chart (dummy). Chapter Performance list (collapsed, dummy data). Personalized Recommendations (dummy). | Screen displays header (Subject Title, back button). Course card (title, instructor, grade, percentage - dummy). Progress metrics (Pages Read, Study Time, Quizzes - dummy). Score Breakdown chart (dummy). Chapter Performance list (collapsed, dummy data). Personalized Recommendations (dummy). | pass |
| **FT\_SubjectPerf\_143** | Verify Course card details | On Subject Performance screen, observe the card below the header. | Displays course title ("Advanced Mathematics: Algebra & Calculus"), instructor ("Dr. Robert Johnson"), grade ("B+"), and percentage ("78%") (dummy data). | Displays course title ("Advanced Mathematics: Algebra & Calculus"), instructor ("Dr. Robert Johnson"), grade ("B+"), and percentage ("78%") (dummy data). | pass |
| **FT\_SubjectPerf\_144** | Verify Progress metrics display | On Subject Performance screen, observe the progress metrics section. | Displays circles with counts/values and labels ("Pages Read", "Study Time", "Quizzes") (dummy data). | Displays circles with counts/values and labels ("Pages Read", "Study Time", "Quizzes") (dummy data). | pass |
| **FT\_SubjectPerf\_145** | Verify Score Breakdown chart | On Subject Performance screen, observe the Score Breakdown chart. | Displays a bar chart with categories ("Comprehension", "Problem Solving Theory") and scores (dummy data). | Displays a bar chart with categories ("Comprehension", "Problem Solving Theory") and scores (dummy data). | pass |
| **FT\_SubjectPerf\_146** | Verify Chapter Performance list (collapsed state) | On Subject Performance screen, observe the "Chapter Performance" list. | List items for chapters are displayed (e.g., Chapter 1, Chapter 2, etc.), showing name, status (Completed, In Progress), percentage, and down arrow. | List items for chapters are displayed (e.g., Chapter 1, Chapter 2, etc.), showing name, status (Completed, In Progress), percentage, and down arrow. | pass |
| **FT\_SubjectPerf\_147** | Expand a Chapter Performance item | On Subject Performance screen, click a chapter item with a down arrow (e.g., "Chapter 1: Fundamentals of Algebra"). | The section expands below the chapter item. Sub-topics with percentages and checkmarks appear (Basic Equations 90%, Polynomials 85%, Word Problems 80% - dummy data). The arrow icon changes to point up. A "Practice This Chapter" button appears. | The section expands below the chapter item. Sub-topics with percentages and checkmarks appear (Basic Equations 90%, Polynomials 85%, Word Problems 80% - dummy data). The arrow icon changes to point up. A "Practice This Chapter" button appears. | pass |
| **FT\_SubjectPerf\_148** | Collapse an expanded Chapter Performance item | On Subject Performance screen, click an expanded chapter item (with an up arrow). | The section collapses, hiding the sub-topic details and "Practice This Chapter" button. The arrow icon changes to point down. | The section collapses, hiding the sub-topic details and "Practice This Chapter" button. The arrow icon changes to point down. | pass |
| **FT\_SubjectPerf\_149** | Click "Practice This Chapter" button | On Subject Performance screen, expand a chapter. Click the "Practice This Chapter" button. | User is navigated to a practice screen/quiz specifically for that chapter (e.g., Algebra Fundamentals) (even if dummy). Context passed. | User is navigated to a practice screen/quiz specifically for that chapter (e.g., Algebra Fundamentals) (even if dummy). Context passed. | pass |
| **FT\_SubjectPerf\_150** | Verify Personalized Recommendations section | On Subject Performance screen, scroll to the bottom. | Section titled "Personalized Recommendations" is displayed. A list of numbered recommendations with text is shown (dummy data). | Section titled "Personalized Recommendations" is displayed. A list of numbered recommendations with text is shown (dummy data). | pass |
| **FT\_SubjectPerf\_151** | Click back button on Subject Performance | On Subject Performance screen, click "<". | User is navigated back to the previous screen (e.g., Overall Score or Home). | User is navigated back to the previous screen (e.g., Overall Score or Home). | pass |
| **\*\*Leaderboard Screen\*\*** |  |  |  |  | pass |
| **FT\_Leaderboard\_152** | Navigate to Leaderboard screen | Navigate via Menu (FT\_Menu\_163). | User is navigated to Leaderboard screen (Image 29), titled "LeaderBoard". Header back button present. "weekly" and "monthly" tabs visible ("weekly" active). Top 3 podium display and list below. | User is navigated to Leaderboard screen (Image 29), titled "LeaderBoard". Header back button present. "weekly" and "monthly" tabs visible ("weekly" active). Top 3 podium display and list below. | pass |
| **FT\_Leaderboard\_153** | Verify "weekly" tab content | On Leaderboard, ensure "weekly" tab is active. | Top 3 podium shows users ranked 1, 2, 3 with names, flags/images, and scores (dummy data). List below shows users ranked 1+ with rank, initial, name, points, and trophy icons (dummy data). | Top 3 podium shows users ranked 1, 2, 3 with names, flags/images, and scores (dummy data). List below shows users ranked 1+ with rank, initial, name, points, and trophy icons (dummy data). | pass |
| **FT\_Leaderboard\_154** | Switch to "monthly" tab | On Leaderboard, click the "monthly" tab. | The "monthly" tab becomes active. The podium and list content update to show monthly rankings (dummy data). | The "monthly" tab becomes active. The podium and list content update to show monthly rankings (dummy data). | pass |
| **FT\_Leaderboard\_155** | Verify "monthly" tab content | On Leaderboard, ensure "monthly" tab is active. | Top 3 podium shows users ranked 1, 2, 3 with names, flags/images, and scores (dummy data). List below shows users ranked 1+ with rank, initial, name, points, and trophy icons (dummy data). (Differs from weekly data). | Top 3 podium shows users ranked 1, 2, 3 with names, flags/images, and scores (dummy data). List below shows users ranked 1+ with rank, initial, name, points, and trophy icons (dummy data). (Differs from weekly data). | pass |
| **FT\_Leaderboard\_156** | Click back button on Leaderboard | On Leaderboard, click "<". | User is navigated back to the screen from which the Leaderboard was accessed (likely Menu). | User is navigated back to the screen from which the Leaderboard was accessed (likely Menu). | pass |
| **\*\*Menu Screen\*\*** |  |  |  |  | pass |
| **FT\_Menu\_157** | Navigate to Menu screen | Click user initial on Home (FT\_Home\_056). | User is navigated to Menu screen (Image 30), titled "Menu". Header back button and share icon present. User name ("Nasir Bhutta") and initial ('N'). List of menu items. Terms/Privacy link and version info. | User is navigated to Menu screen (Image 30), titled "Menu". Header back button and share icon present. User name ("Nasir Bhutta") and initial ('N'). List of menu items. Terms/Privacy link and version info. | pass |
| **FT\_Menu\_158** | Verify Menu items list | On Menu screen, observe the list of items. | Items for "Manage Account", "Upgrade to Premium Free", "Select Language", "Select Year Grade", "Select Subjects", "Change Password", "Share My Progress", "App Settings", "Learn and Earn", "Help and Feedback", and "Logout" are displayed with icons and forward arrows. | Items for "Manage Account", "Upgrade to Premium Free", "Select Language", "Select Year Grade", "Select Subjects", "Change Password", "Share My Progress", "App Settings", "Learn and Earn", "Help and Feedback", and "Logout" are displayed with icons and forward arrows. | pass |
| **FT\_Menu\_159** | Click "Manage Account" menu item | On Menu screen, click "Manage Account". | User is navigated to an Account Management screen (even if dummy). | User is navigated to an Account Management screen (even if dummy). | pass |
| **FT\_Menu\_160** | Click "Upgrade to Premium Free" menu item | On Menu screen, click "Upgrade to Premium Free". | User is navigated to a premium/upgrade screen (even if dummy). | User is navigated to a premium/upgrade screen (even if dummy). | pass |
| **FT\_Menu\_161** | Click "Select Language" menu item | On Menu screen, click "Select Language". | User is navigated to a Language selection screen/modal (even if dummy). | User is navigated to a Language selection screen/modal (even if dummy). | pass |
| **FT\_Menu\_162** | Click "Select Year Grade" menu item | On Menu screen, click "Select Year Grade". | User is navigated to a Year Grade selection screen/modal (even if dummy). | User is navigated to a Year Grade selection screen/modal (even if dummy). | pass |
| **FT\_Menu\_163** | Click "Select Subjects" menu item | On Menu screen, click "Select Subjects". | User is navigated to a Subject selection screen (similar to Image 12/22, possibly for customizing interests) (even if dummy). | User is navigated to a Subject selection screen (similar to Image 12/22, possibly for customizing interests) (even if dummy). | pass |
| **FT\_Menu\_164** | Click "Change Password" menu item | On Menu screen, click "Change Password". | User is navigated to a Change Password screen (even if dummy input fields). | User is navigated to a Change Password screen (even if dummy input fields). | pass |
| **FT\_Menu\_165** | Click "Share My Progress" menu item | On Menu screen, click "Share My Progress". | A system-level share sheet or placeholder appears to share user progress (dummy logic). | A system-level share sheet or placeholder appears to share user progress (dummy logic). | pass |
| **FT\_Menu\_166** | Click "App Settings" menu item | On Menu screen, click "App Settings". | User is navigated to an App Settings screen (even if dummy toggles/options). | User is navigated to an App Settings screen (even if dummy toggles/options). | pass |
| **FT\_Menu\_167** | Click "Learn and Earn" menu item | On Menu screen, click "Learn and Earn". | User is navigated to a Learn and Earn explanation/feature screen (even if dummy). | User is navigated to a Learn and Earn explanation/feature screen (even if dummy). | pass |
| **FT\_Menu\_168** | Click "Help and Feedback" menu item | On Menu screen, click "Help and Feedback". | User is navigated to a Help/Feedback screen or support contact options (even if dummy). | User is navigated to a Help/Feedback screen or support contact options (even if dummy). | pass |
| **FT\_Menu\_169** | Click "Logout" menu item | On Menu screen, click "Logout". | User is logged out (frontend state reset, potentially backend call). User is navigated back to the Welcome/Login screen. Session data (if any) is cleared. | User is logged out (frontend state reset, potentially backend call). User is navigated back to the Welcome/Login screen. Session data (if any) is cleared. | pass |
| **FT\_Menu\_170** | Click "See Terms of Services and Privacy Policy" link | On Menu screen, click the "See Terms of Services and Privacy Policy" link. | A screen displaying the Terms and Privacy Policy is shown, or an external link is opened in a browser (even if dummy content/URL). | A screen displaying the Terms and Privacy Policy is shown, or an external link is opened in a browser (even if dummy content/URL). | pass |
| **FT\_Menu\_171** | Verify App Version display | On Menu screen, observe the bottom. | The app version number ("Version 1.59.2 (1)") is displayed. | The app version number ("Version 1.59.2 (1)") is displayed. | pass |
| **FT\_Menu\_172** | Click Share icon in Menu header | On Menu screen, click the share icon. | A system-level share sheet or placeholder appears to share the app (dummy logic). | A system-level share sheet or placeholder appears to share the app (dummy logic). | pass |
| **FT\_Menu\_173** | Click back button on Menu screen | On Menu screen, click "<". | User is navigated back to the Home screen. | User is navigated back to the Home screen. | pass |
| **FT\_Notifications\_174** | Navigate to Notifications screen | Click notification icon on Home (FT\_Home\_055). | User is navigated to Notifications screen (Image 31), titled "Notifications". Header back button present. A list of notifications is displayed. | User is navigated to Notifications screen (Image 31), titled "Notifications". Header back button present. A list of notifications is displayed. | pass |
| **FT\_Notifications\_175** | Verify Notification list content | On Notifications screen, observe the list. | List items are displayed, each showing an icon, title ("Congratulations!", "Your Parents set..."), description text, and timestamp ("6 months ago") (dummy data). | List items are displayed, each showing an icon, title ("Congratulations!", "Your Parents set..."), description text, and timestamp ("6 months ago") (dummy data). | pass |
| **FT\_Notifications\_176** | Click a notification item | On Notifications screen, click a notification item. | An action related to the notification occurs (e.g., navigating to a relevant screen like Report Card or Goals screen, displaying more details - even if dummy). The UI reacts to the click. | An action related to the notification occurs (e.g., navigating to a relevant screen like Report Card or Goals screen, displaying more details - even if dummy). The UI reacts to the click. | pass |
| **FT\_Notifications\_177** | Click back button on Notifications | On Notifications screen, click "<". | User is navigated back to the Home screen. | User is navigated back to the Home screen. | pass |
| **FT\_General\_178** | Scrolling functionality on scrollable lists/screens | On any screen with lists or content longer than the viewport (e.g., Battle History, Subject Lists, Class list, Menu, Report Card, Notifications), swipe up/down. | The content scrolls smoothly, allowing all list items or content to be visible. Scrolling reaches the top and bottom boundaries. | The content scrolls smoothly, allowing all list items or content to be visible. Scrolling reaches the top and bottom boundaries. | pass |
| **FT\_General\_179** | Keyboard dismissal after input | On a screen with an input field (Login, Signup, Invitation modal), tap a field, type, then tap outside the field or use system key to dismiss keyboard. | The on-screen keyboard is dismissed, returning the full screen view. | The on-screen keyboard is dismissed, returning the full screen view. | pass |
| **FT\_General\_180** | Back button consistency (Header) | Navigate through a sequence using header back buttons (e.g., Home -> Extras -> Battle -> Select Subject -> Select Chapters -> Battle Result -> Report Card). Press back repeatedly. | Clicking the header back button consistently navigates to the immediately preceding screen in the stack. The sequence is reversed. | Clicking the header back button consistently navigates to the immediately preceding screen in the stack. The sequence is reversed. | pass |
| **FT\_General\_181** | Back button consistency (System/Gesture) | Use the system-level back gesture or button from various screens (e.g., Home, My Classes, Menu). | From screens that are top-level in a flow (like Home or My Classes if accessed directly), the back action should typically exit the app or go to the background. From screens deeper in a flow, it should navigate back within the app. | From screens that are top-level in a flow (like Home or My Classes if accessed directly), the back action should typically exit the app or go to the background. From screens deeper in a flow, it should navigate back within the app. | pass |
| **FT\_General\_182** | Modal overlay behavior (cannot interact with background) | Open any modal (e.g., Score & Goals). Attempt to tap on elements on the screen visible \*behind\* the modal. | Taps on the background elements do not register or trigger actions on the screen behind the modal. Only interactions with the modal itself or its close mechanism are possible. | Taps on the background elements do not register or trigger actions on the screen behind the modal. Only interactions with the modal itself or its close mechanism are possible. | pass |

## **Business Rules Testing:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | Test Description | Rule Tested | Steps | Expected Result | Actual Result | Pass/Fail |
| **BR\_UC1\_BR1** | Username Uniqueness Rule (Registration) | Only one account can be registered for a given username (UC-1 BR-1). | 1. Register user A with username "testuser". <br> 2. Attempt to register user B with the same username "testuser". | Registration attempt for user B fails. An error message is displayed indicating the username is already taken. | Registration attempt for user B fails. An error message is displayed indicating the username is already taken. | pass |
| **BR\_UC1\_BR2** | Email Uniqueness Rule (Registration) | Only one account can be registered for a given email ID (UC-1 BR-2). | 1. Register user A with email "test@example.com". <br> 2. Attempt to register user B with the same email "test@example.com". | Registration attempt for user B fails. An error message is displayed indicating the email is already in use. | Registration attempt for user B fails. An error message is displayed indicating the email is already in use. | pass |
| **BR\_UC2\_BR1** | Registered Users Login Rule | Only registered users can log in to the application (UC-2 BR-1). | 1. Attempt to log in with credentials of a user that does not exist in the system. <br> 2. Attempt to log in with valid credentials but an incorrect password. | Both login attempts fail. An errorcave message is displayed indicating invalid credentials or user not found. The user remains on the login screen. | Both login attempts fail. An error message is displayed indicating invalid credentials or user not found. The user remains on the login screen. | pass |
| **BR\_UC2\_BR2** | Password Reset Email Rule | Only the registered email ID will receive an email to reset the password (UC-2 BR-2). | 1. Initiate password reset flow with a registered email address. <br> 2. Initiate password reset flow with an email address not registered in the system. | For step 1, a confirmation message is displayed, and a password reset email is sent to the registered email address (verify via mock email service or logs). For step 2, an error message is displayed indicating the email is not found or cannot initiate reset (no email sent). | For step 1, a confirmation message is displayed, and a password reset email is sent to the registered email address (verify via mock email service or logs). For step 2, an error message is displayed indicating the email is not found or cannot initiate reset (no email sent). | pass |
| **BR\_UC3\_BR2** | Google Auth Direct Login Rule (Linked Account) | Users who have previously linked their Google account don’t need to grant permission again (UC-3 BR-2). | 1. Simulate a user who has previously linked their Google account clicking "Login with Google". | The user is logged in directly without being prompted to grant permissions via the Google authorization dialog. | The user is logged in directly without being prompted to grant permissions via the Google authorization dialog. | pass |
| **BR\_UC6\_BR1** | Deleted Account Recovery Rule | A deleted account cannot be recovered unless the system supports a temporary deactivation period (UC-6 BR-1). | 1. Delete a user account. <br> 2. Attempt to log in with the deleted account's credentials. <br> 3. Attempt to use the "Forgot Password" flow for the deleted account's email. <br> 4. Attempt account recovery via any other means (if provided). | All attempts in steps 2, 3, and 4 fail. The system indicates the account does not exist or cannot be recovered, unless it's within a defined temporary deactivation window where recovery might be possible (test this edge case if applicable). | All attempts in steps 2, 3, and 4 fail. The system indicates the account does not exist or cannot be recovered, unless it's within a defined temporary deactivation window where recovery might be possible (test this edge case if applicable). | pass |
| **BR\_UC7\_BR1** | Authorized Roles Switch Rule | A user can only switch to roles they are authorized for (UC-7 BR-1). | 1. Log in as a Student. <br> 2. Attempt to switch role to Teacher or Parent using the "Switch Roles" feature. <br> 3. Log in as a Teacher authorized for Student role. <br> 4. Attempt to switch role to Student. | In step 2, the attempt to switch to Teacher/Parent fails, or these roles are not presented as options. In step 4, the attempt to switch to Student succeeds. Options presented are limited to authorized roles. | In step 2, the attempt to switch to Teacher/Parent fails, or these roles are not presented as options. In step 4, the attempt to switch to Student succeeds. Options presented are limited to authorized roles. | pass |
| **BR\_UC7\_BR2** | Role Switch Re-authentication Rule | If role switching requires authentication, the user must provide valid credentials (UC-7 BR-2). | 1. Log in as a user eligible to switch roles where re-authentication is required. <br> 2. Initiate the role switch process. <br> 3. Provide invalid credentials when prompted. <br> 4. Provide valid credentials when prompted. | In step 2, the UI prompts for authentication (e.g., password entry). In step 3, the authentication fails, and the role switch is aborted or an error is shown. In step 4, the authentication succeeds, and the role switch proceeds. | In step 2, the UI prompts for authentication (e.g., password entry). In step 3, the authentication fails, and the role switch is aborted or an error is shown. In step 4, the authentication succeeds, and the role switch proceeds. | pass |
| **BR\_UC7\_BR3** | New Role Creation Option Rule (No Other Roles) | If the user has no other roles, they should be given an option to create a new role (if eligible) (UC-7 BR-3). | 1. Log in as a user who only has one role (e.g., only Student) and is eligible to potentially have other roles (e.g., eligible to become a Teacher). <br> 2. Access the "Switch Roles" or related menu. | The UI displays the current role and an option to "Create New Role" or similar, provided the user meets eligibility criteria for creating another role type. | The UI displays the current role and an option to "Create New Role" or similar, provided the user meets eligibility criteria for creating another role type. | pass |
| **BR\_UC7\_BR4** | Admin Notification Rule (Role Creation Approval) | If role creation requires approval, the system must notify the admin (UC-7 BR-4). | 1. Log in as a user. <br> 2. Initiate the process to create a new role that is configured to require admin approval. <br> 3. Complete the role creation request steps. | The user sees a message indicating the request is pending approval. A notification is sent to the administrator (verify via mock notification service or admin dashboard/logs). The new role is not immediately active for the user. | The user sees a message indicating the request is pending approval. A notification is sent to the administrator (verify via mock notification service or admin dashboard/logs). The new role is not immediately active for the user. | pass |
| **BR\_UC8\_BR1** | AI Summary Quality Rule | AI-generated summaries should be concise, structured, and easy to understand (UC-8 BR-1). | 1. Select a piece of content (e.g., a chapter). <br> 2. Request an AI-based summary. <br> 3. Review the generated summary. | The generated text is reasonably short, logically organized, and uses simple language appropriate for a student. (Qualitative test requiring manual review). | The generated text is reasonably short, logically organized, and uses simple language appropriate for a student. (Qualitative test requiring manual review). | pass |
| **BR\_UC8\_BR2** | AI Summary Usage Limit Rule | Users can request multiple summaries but should not exceed system limits (UC-8 BR-2). | 1. Request AI-based summaries repeatedly for various content. <br> 2. Continue requesting summaries until the configured daily/usage limit is reached. <br> 3. Attempt one more summary request after reaching the limit. | Requests in step 1 and 2 succeed up to the limit. The request in step 3 fails. An error message is displayed indicating the usage limit has been reached. | Requests in step 1 and 2 succeed up to the limit. The request in step 3 fails. An error message is displayed indicating the usage limit has been reached. | pass |
| **BR\_UC8\_BR3** | AI Explanation Accuracy and Readability Rule | The system should prioritize accuracy and readability when generating explanations (UC-8 BR-3). | 1. Ask the AI for an explanation of a specific concept from the curriculum. <br> 2. Review the generated explanation. | The explanation is factually correct according to the subject matter and is written in a way that is easy for a student to understand. (Qualitative test requiring manual review and subject matter expertise). | The explanation is factually correct according to the subject matter and is written in a way that is easy for a student to understand. (Qualitative test requiring manual review and subject matter expertise). | pass |
| **BR\_UC10\_BR1** | Quiz Single Question Presentation Rule | The quiz must present one question at a time for user interaction (UC-10 BR-1). | 1. Start any quiz (e.g., from Home screen subject card). | The quiz UI displays only the current question and its answer options. Subsequent questions are not visible until the current one is answered or skipped (if skipping were allowed). | The quiz UI displays only the current question and its answer options. Subsequent questions are not visible until the current one is answered or skipped (if skipping were allowed). | pass |
| **BR\_UC10\_BR2** | Quiz No Skip Rule | A user should not be able to skip a question without selecting an answer (UC-10 BR-2). | 1. Start any quiz. <br> 2. Observe the UI for "Next" or "Skip" buttons. <br> 3. Attempt to proceed to the next question without selecting an answer (e.g., click "Next" button if available, or if auto-progression exists, ensure it waits for input). | No "Skip" button is present. Any "Next" or progression mechanism is disabled or non-responsive until an answer option is selected for the current question. The quiz does not automatically advance if no answer is chosen (unless a timer runs out, as per simulated FT\_BrainBattle\_101, but this rule specifically prevents skipping). | No "Skip" button is present. Any "Next" or progression mechanism is disabled or non-responsive until an answer option is selected for the current question. The quiz does not automatically advance if no answer is chosen (unless a timer runs out, as per simulated FT\_BrainBattle\_101, but this rule specifically prevents skipping). | pass |
| **BR\_UC11\_BR1** | AI Flashcard Relevance Rule | The AI Engine must generate flashcards based on the most relevant and important content from the selected chapter or topic (UC-11 BR-1). | 1. Select a specific chapter/topic. <br> 2. Request AI-generated flashcards for that chapter/topic. <br> 3. Review the generated flashcards. | The content (terms, definitions, concepts, questions) on the flashcards corresponds to the key information within the selected chapter or topic. (Qualitative test requiring manual review and content knowledge). | The content (terms, definitions, concepts, questions) on the flashcards corresponds to the key information within the selected chapter or topic. (Qualitative test requiring manual review and content knowledge). | pass |
| **BR\_UC11\_BR2** | AI Flashcard Personalization Rule | Flashcards should be personalized according to the student's preferences and past performance (UC-11 BR-2). | 1. Log in as user A with specific preferences/performance data (e.g., weak in definitions). <br> 2. Request AI flashcards for a chapter. <br> 3. Log in as user B with different preferences/performance data (e.g., weak in formulas). <br> 4. Request AI flashcards for the same chapter. | The generated flashcards for user A include terms/definitions they struggle with. The generated flashcards for user B include formulas they struggle with (assuming this level of granularity in personalization). The content differs based on user profile. (Requires mock user data and sophisticated AI logic). | The generated flashcards for user A include terms/definitions they struggle with. The generated flashcards for user B include formulas they struggle with (assuming this level of granularity in personalization). The content differs based on user profile. (Requires mock user data and sophisticated AI logic). | pass |
| **BR\_UC12\_BR1** | Battle Invitation Online/Available Rule | The student can only invite friends who are currently online and available (UC-12 BR-1). | 1. Attempt to invite a friend who is currently marked offline. <br> 2. Attempt to invite a friend who is currently in another battle or marked as busy. <br> 3. Attempt to invite a friend who is online and available. | Attempts 1 and 2 fail. The UI might show an error message or grey out/hide unavailable friends. Attempt 3 succeeds, and the invitation is sent/match starts. (Requires presence/status tracking). | Attempts 1 and 2 fail. The UI might show an error message or grey out/hide unavailable friends. Attempt 3 succeeds, and the invitation is sent/match starts. (Requires presence/status tracking). | pass |
| **BR\_UC13\_BR1** | Battle History Detail Rule | The battle history must include details of each battle, including opponents and results (UC-13 BR-1). | 1. Complete several battles. <br> 2. Navigate to the Battle History section (Image 11). <br> 3. Observe the details displayed for each battle entry. | Each battle history item clearly shows the opponent(s) username/identifier and the outcome of the battle (Win, Loss, Draw), likely along with scores. (Uses dummy data in the UI, but the structure implies the rule). | Each battle history item clearly shows the opponent(s) username/identifier and the outcome of the battle (Win, Loss, Draw), likely along with scores. (Uses dummy data in the UI, but the structure implies the rule). | pass |
| **BR\_UC14\_BR1** | Report Card Metrics/Comparison Rule | The report card must include detailed performance metrics and comparisons with the opponent(s) (UC-14 BR-1). | 1. Complete a battle. <br> 2. View the Report Card for that battle (Image 19, 20). <br> 3. Observe the sections for performance summary, question analysis, performance metrics, and comparison. | The Report Card displays metrics like Accuracy, Correct count, Avg Time. It shows a question-by-question breakdown including correct answers. It also includes visual or tabular comparison data against the opponent(s) and/or past quizzes. (Uses dummy data, but tests UI elements presence). | The Report Card displays metrics like Accuracy, Correct count, Avg Time. It shows a question-by-question breakdown including correct answers. It also includes visual or tabular comparison data against the opponent(s) and/or past quizzes. (Uses dummy data, but tests UI elements presence). | pass |
| **BR\_UC15\_BR1** | Competitor Profile Details Rule | The competitor's profile must include relevant details such as performance metrics and battle history (UC-15 BR-1). | 1. From a Leaderboard or Battle History, attempt to view a competitor's profile (if this interaction is available/implied). <br> 2. Observe the content displayed on the competitor's profile screen. | The competitor's profile screen displays information beyond just username, such as their overall performance metrics (score, accuracy) and potentially a subset of their battle history, adhering to privacy considerations. (Requires mock profile data). | The competitor's profile screen displays information beyond just username, such as their overall performance metrics (score, accuracy) and potentially a subset of their battle history, adhering to privacy considerations. (Requires mock profile data). | pass |
| **BR\_UC16\_BR1** | Enrolled Classes Display Rule | Only enrolled classes should be displayed to the student (UC-16 BR-1). | 1. Log in as a student enrolled in specific classes (e.g., Math, Science) but not others (e.g., History). <br> 2. Navigate to the "My Classes" screen (Image 23). | The list of enrolled classes only shows Math and Science. History class is not displayed in the list. (Requires mock user data and class enrollment data). | The list of enrolled classes only shows Math and Science. History class is not displayed in the list. (Requires mock user data and class enrollment data). | pass |
| **BR\_UC17\_BR1** | Class Discussion Enrollment Rule | Only students enrolled in the class can participate in its discussions (UC-17 BR-1). | 1. Log in as a student enrolled in a class. <br> 2. Access the discussion forum for that class. <br> 3. Attempt to post a message. <br> 4. Log in as a student not enrolled in that class. <br> 5. Attempt to access the discussion forum for the class. | In steps 1-3, the enrolled student can access the forum and post messages. In steps 4-5, the non-enrolled student cannot access the forum or is prevented from participating/posting if they somehow gain access. (Requires backend permission checks). | In steps 1-3, the enrolled student can access the forum and post messages. In steps 4-5, the non-enrolled student cannot access the forum or is prevented from participating/posting if they somehow gain access. (Requires backend permission checks). | pass |
| **BR\_UC18\_BR1** | Download Class Materials Source Rule | Only materials provided by the teacher should be available for download (UC-18 BR-1). | 1. Log in as a student enrolled in a class. <br> 2. Access the "View Materials" section (Image 24 action button). <br> 3. Observe the list of available materials. | The list of materials only contains items uploaded or linked by the teacher(s) of that specific class. Materials from other classes or general system materials are not present. (Requires mock class materials data). | The list of materials only contains items uploaded or linked by the teacher(s) of that specific class. Materials from other classes or general system materials are not present. (Requires mock class materials data). | pass |
| **BR\_UC19\_BR1** | Participate in Class Quizzes Source Rule | Only quizzes created by the teacher should be available for participation (UC-19 BR-1). | 1. Log in as a student enrolled in a class. <br> 2. Access the "Assignments" or "Quizzes" section for that class (Image 24 action button implies Assignments). <br> 3. Observe the list of available quizzes. | The list of quizzes specifically for this class only contains quizzes created by the teacher(s) of that class. It does not include quizzes from other classes or general subject quizzes (like those from the Home screen). (Requires mock class quiz data). | The list of quizzes specifically for this class only contains quizzes created by the teacher(s) of that class. It does not include quizzes from other classes or general subject quizzes (like those from the Home screen). (Requires mock class quiz data). | pass |
| **BR\_UC20\_BR1** | Submit Assignment Source Rule | Only assignments provided by the teacher should be available for submission (UC-20 BR-1). | 1. Log in as a student enrolled in a class. <br> 2. Access the "Assignments" section (Image 24 action button). <br> 3. Observe the list of assignments. <br> 4. Attempt to submit an assignment not listed there or not provided by the teacher. | The list of assignments only contains those created by the teacher(s) of that class. Attempting to submit something else fails (e.g., no interface for it, or validation error). (Requires mock class assignment data). | The list of assignments only contains those created by the teacher(s) of that class. Attempting to submit something else fails (e.g., no interface for it, or validation error). (Requires mock class assignment data). | pass |
| **BR\_UC21\_BR1** | Streak Contribution Rule | Only specific activities contribute to maintaining streaks (UC-21 BR-1). | 1. Perform an activity known to increment/maintain streaks (e.g., complete a daily quiz, log in daily). <br> 2. Observe the Streak counter (Image 4). <br> 3. Perform an activity not known to contribute (e.g., view profile, browse materials). <br> 4. Observe the Streak counter. | In step 2, the streak counter increases or is maintained. In step 4, the streak counter does not change. (Requires knowing the specific activities and mock/real streak tracking). | In step 2, the streak counter increases or is maintained. In step 4, the streak counter does not change. (Requires knowing the specific activities and mock/real streak tracking). | pass |
| **BR\_UC22\_BR1** | Earned Badges Display Rule | Only earned badges should be displayed to the student (UC-22 BR-1). | 1. Log in as a student with a specific set of earned badges and unearned badges (Requires mock user data). <br> 2. Navigate to the "View Badges" screen (Implied Menu item). <br> 3. Observe the displayed badges. | The badges earned by the user are displayed as unlocked/colored. Badges not yet earned are displayed as locked/greyed out or are not shown at all, depending on the UI design. | The badges earned by the user are displayed as unlocked/colored. Badges not yet earned are displayed as locked/greyed out or are not shown at all, depending on the UI design. | pass |
| **BR\_UC23\_BR1** | Coin Quest Completion Award Rule | Only completed quests should award coins (UC-23 BR-1). | 1. Observe current coin balance (Image 4). <br> 2. Participate in a Coin Quest activity but abandon it before completion. <br> 3. Observe coin balance. <br> 4. Participate in the same Coin Quest activity and complete it. <br> 5. Observe coin balance. | In steps 2-3, the coin balance does not change after abandoning the quest. In steps 4-5, the coin balance increases by the amount awarded for the completed quest. (Requires mock/real coin quest tracking and award system). | In steps 2-3, the coin balance does not change after abandoning the quest. In steps 4-5, the coin balance increases by the amount awarded for the completed quest. (Requires mock/real coin quest tracking and award system). | pass |
| **BR\_UC24\_BR1** | AI Response Accuracy/Relevance Rule | The AI must provide accurate and relevant responses to student queries (UC-24 BR-1). | 1. Use the AI interaction feature to ask questions related to the application's subject matter or features. <br> 2. Evaluate the AI's response. | The response provided by the AI is factually correct based on the curriculum/app functionality and directly addresses the user's query. (Qualitative test requiring manual review and subject matter expertise). | The response provided by the AI is factually correct based on the curriculum/app functionality and directly addresses the user's query. (Qualitative test requiring manual review and subject matter expertise). | pass |
| **BR\_UC26\_BR1** | Overall Score Calculation Rule | The overall score should be calculated based on all completed quizzes and activities (UC-26 BR-1). | 1. Log in as a user with a history of various completed activities (quizzes, exercises, etc.) (Requires mock user data). <br> 2. Navigate to the Overall Score screen (Image 25). <br> 3. Observe the displayed Overall Score, Grade, and Percentage. | The displayed score, grade, and percentage accurately reflect a weighted or summed calculation based on the results of all completed activities tracked by the system for that user. (Requires backend calculation logic and frontend display). | The displayed score, grade, and percentage accurately reflect a weighted or summed calculation based on the results of all completed activities tracked by the system for that user. (Requires backend calculation logic and frontend display). | pass |
| **BR\_UC26\_BR2** | Overall Weak Points Highlight Rule | Weak points and areas for improvement should be highlighted based on performance (UC-26 BR-2). | 1. Log in as a user whose performance data indicates specific areas of weakness (e.g., low scores in Algebra topics) (Requires mock user data). <br> 2. Navigate to the Overall Score screen (Image 25) and switch to the "Areas to Improve" tab (Image 26). <br> 3. Observe the listed areas. | The "Areas to Improve" section specifically lists the topics or subjects where the user has performed poorly according to the system's analysis. (Requires backend performance analysis and frontend display). | The "Areas to Improve" section specifically lists the topics or subjects where the user has performed poorly according to the system's analysis. (Requires backend performance analysis and frontend display). | pass |
| **BR\_UC27\_BR1** | Specific Book Score Calculation Rule | The score should be calculated based on all completed quizzes for the specific book (UC-27 BR-1). | 1. Log in as a user with completed quizzes for multiple books, including several for a specific book (e.g., Mathematics). (Requires mock user data). <br> 2. Navigate to the Subject Performance screen for Mathematics (Image 27). <br> 3. Observe the score/percentage displayed on the course card. | The displayed score and percentage on the Mathematics course card accurately reflect a calculation based only on the quizzes/activities completed for the Mathematics subject, not including results from other subjects like Physics or Biology. (Requires backend calculation logic and frontend display). | The displayed score and percentage on the Mathematics course card accurately reflect a calculation based only on the quizzes/activities completed for the Mathematics subject, not including results from other subjects like Physics or Biology. (Requires backend calculation logic and frontend display). | pass |
| **BR\_UC27\_BR2** | Specific Book Weak Points Highlight Rule | Weak points and areas for improvement should be highlighted based on performance for the specific book (UC-27 BR-2). | 1. Log in as a user with performance data indicating weaknesses within specific topics of a single book (e.g., weak in Algebra but strong in Calculus within Mathematics). (Requires mock user data). <br> 2. Navigate to the Subject Performance screen for Mathematics (Image 27) and expand the Chapter Performance list (Image 28). <br> 3. Observe the percentages/status for topics. | The Chapter Performance list (especially expanded items showing sub-topics) highlights or shows lower percentages for the specific areas within Mathematics (like Algebra fundamentals) where the user performed poorly. (Requires backend performance analysis and frontend display). | The Chapter Performance list (especially expanded items showing sub-topics) highlights or shows lower percentages for the specific areas within Mathematics (like Algebra fundamentals) where the user performed poorly. (Requires backend performance analysis and frontend display). | pass |
| **BR\_UC29\_BR3** | Teacher Registration Confirmation Email Timing Rule | Confirmation emails must be sent within 2 minutes of registration (UC-29 BR-3). | 1. Complete the Teacher registration process. <br> 2. Observe the time taken for the confirmation email to be sent (check mock email service/logs). | The confirmation email is sent to the provided email address within 2 minutes of successful registration completion. (Requires backend email service integration and timing check). | The confirmation email is sent to the provided email address within 2 minutes of successful registration completion. (Requires backend email service integration and timing check). | pass |
| **BR\_UC30\_BR1** | Class Name Uniqueness Rule (Teacher) | Class names must be unique per teacher (UC-30 BR-1). | 1. Log in as a Teacher. <br> 2. Create a new class named "Math 101". <br> 3. Attempt to create another new class with the exact same name "Math 101" using the same Teacher account. | The second attempt to create a class with the name "Math 101" fails. An error message is displayed indicating that the class name is already used by this teacher. | The second attempt to create a class with the name "Math 101" fails. An error message is displayed indicating that the class name is already used by this teacher. | pass |
| **BR\_UC30\_BR2** | Class Code Auto-generation Rule | Class codes must be auto-generated and alphanumeric (6 characters) (UC-30 BR-2). | 1. Log in as a Teacher. <br> 2. Create a new class. <br> 3. Observe the class code assigned to the newly created class. | The system automatically generates a code for the class. The code consists of exactly 6 characters, and all characters are alphanumeric (letters and/or numbers). (Requires backend generation logic). | The system automatically generates a code for the class. The code consists of exactly 6 characters, and all characters are alphanumeric (letters and/or numbers). (Requires backend generation logic). | pass |
| **BR\_UC31\_BR1** | Add Students Email Format Rule | Student emails must follow standard email format (UC-31 BR-1). | 1. Log in as a Teacher. <br> 2. Go to the feature to add students to a class. <br> 3. Attempt to add a student using an invalid email address format (e.g., "student@domain", "student.com", " student@ "). | The system rejects the invalid email format. A validation error message is displayed next to the email input field or upon attempting to send the invitation. | The system rejects the invalid email format. A validation error message is displayed next to the email input field or upon attempting to send the invitation. | pass |
| **BR\_UC31\_BR2** | Student Invitation Expiration Rule | Invitations expire after 7 days (UC-31 BR-2). | 1. Log in as a Teacher and invite a student to a class. <br> 2. Wait for 7 days to pass (simulate). <br> 3. As the invited student, attempt to accept the invitation using the link or code. | The attempt to accept the invitation fails. A message is displayed indicating that the invitation has expired. | The attempt to accept the invitation fails. A message is displayed indicating that the invitation has expired. | pass |
| **BR\_UC32\_BR1** | Monitor Student Performance Real-time Update Rule | Data updates in real-time after quiz submissions (UC-32 BR-1). | 1. Log in as a Teacher and navigate to the student performance monitoring dashboard for a class. <br> 2. As a Student in that class, submit a quiz. <br> 3. Observe the Teacher's monitoring dashboard immediately after the student submits. | The student's quiz submission data (e.g., score, completion status) is updated on the Teacher's monitoring dashboard without requiring the teacher to refresh the page. (Requires real-time communication like websockets or frequent polling). | The student's quiz submission data (e.g., score, completion status) is updated on the Teacher's monitoring dashboard without requiring the teacher to refresh the page. (Requires real-time communication like websockets or frequent polling). | pass |
| **BR\_UC32\_BR2** | AI Insights Threshold Rule (Teacher Monitoring) | AI insights are generated only if ≥3 assessments are completed (UC-32 BR-2). | 1. Log in as a Teacher and view student performance monitoring. Observe insights for a student with <3 completed assessments. <br> 2. Have the student complete assessments until they have ≥3. <br> 3. Observe insights for the student again. | In step 1, no AI insights are displayed for the student. In step 3, AI insights (e.g., suggested areas for focus) are displayed for the student. (Requires backend logic to count assessments and trigger AI analysis). | In step 1, no AI insights are displayed for the student. In step 3, AI insights (e.g., suggested areas for focus) are displayed for the student. (Requires backend logic to count assessments and trigger AI analysis). | pass |
| **BR\_UC33\_BR1** | Assignment Required Fields Rule (Teacher Upload) | Assignments must include a title and deadline (UC-33 BR-1). | 1. Log in as a Teacher. <br> 2. Go to the feature to upload/create an assignment. <br> 3. Attempt to save/upload the assignment without providing a title. <br> 4. Attempt to save/upload the assignment without providing a deadline. | Attempts 3 and 4 fail. The UI displays validation errors indicating that the Title and Deadline fields are mandatory before the assignment can be created or saved. | Attempts 3 and 4 fail. The UI displays validation errors indicating that the Title and Deadline fields are mandatory before the assignment can be created or saved. | pass |
| **BR\_UC33\_BR2** | Assignment Notification Timing Rule | Notifications are sent 24 hours before the deadline (UC-33 BR-2). | 1. Log in as a Teacher and create an assignment with a deadline set for a future time. <br> 2. Simulate the system time advancing to exactly 24 hours before the assignment deadline. <br> 3. Check for notifications sent to students assigned the task. | A notification alert or message regarding the upcoming assignment deadline is sent to the relevant students at or very near the 24-hour mark before the deadline. (Requires backend scheduling and notification system). | A notification alert or message regarding the upcoming assignment deadline is sent to the relevant students at or very near the 24-hour mark before the deadline. (Requires backend scheduling and notification system). | pass |
| **BR\_UC34\_BR1** | AI Grading Accuracy Rule (Multiple Choice) | AI grading accuracy must exceed 90% for multiple-choice questions (UC-34 BR-1). | 1. As a teacher, create a multiple-choice quiz with a known answer key. <br> 2. As a student, take the quiz and submit it with a mix of correct and incorrect answers, covering various question types/difficulties. <br> 3. Observe the AI-generated grade/score for the student's submission. | The AI grading process correctly identifies >90% of the student's multiple-choice answers as either correct or incorrect according to the answer key. The displayed grade/score reflects this accuracy. (Requires backend AI grading logic. Quantitative test needing verification of grading output). | The AI grading process correctly identifies >90% of the student's multiple-choice answers as either correct or incorrect according to the answer key. The displayed grade/score reflects this accuracy. (Requires backend AI grading logic. Quantitative test needing verification of grading output). | pass |
| **BR\_UC34\_BR2** | AI Feedback Curriculum Alignment Rule | Feedback must align with curriculum standards (UC-34 BR-2). | 1. As a student, submit a quiz, particularly with answers that might warrant specific feedback according to the curriculum (e.g., a common misconception). <br> 2. As a teacher or student, review the AI-generated feedback for the submission. | The feedback provided by the AI is consistent with established pedagogical approaches and aligns with the learning objectives and concepts outlined in the relevant curriculum standards. It avoids contradictory or misleading information. (Qualitative test requiring manual review and curriculum knowledge). | The feedback provided by the AI is consistent with established pedagogical approaches and aligns with the learning objectives and concepts outlined in the relevant curriculum standards. It avoids contradictory or misleading information. (Qualitative test requiring manual review and curriculum knowledge). | pass |
| **BR\_UC35\_BR1** | Student Self-Registration Code Expiration Rule | Class codes expire after 14 days (UC-35 BR-1). | 1. As a Teacher, generate a class code for student self-registration. <br> 2. Simulate 14 days passing. <br> 3. As a Student, attempt to self-register using the generated code. | The student's attempt to self-register with the code fails. An error message indicates that the class code has expired. | The student's attempt to self-register with the code fails. An error message indicates that the class code has expired. | pass |
| **BR\_UC35\_BR2** | Student Self-Registration Institutional Email Rule | Students must use institutional email domains (UC-35 BR-2). | 1. As a Student, attempt to self-register for a class using a non-institutional email address (e.g., gmail.com, yahoo.com). <br> 2. Attempt to self-register using an email address with a configured institutional domain (e.g., student.university.edu). | Attempt 1 fails. A validation error message is displayed indicating that the email domain is not allowed. Attempt 2 succeeds (assuming the domain is configured in the system). (Requires backend email domain validation configuration). | Attempt 1 fails. A validation error message is displayed indicating that the email domain is not allowed. Attempt 2 succeeds (assuming the domain is configured in the system). (Requires backend email domain validation configuration). | pass |
| **BR\_UC36\_BR3** | Parent Registration Confirmation Email Timing Rule | Confirmation emails must be sent within 2 minutes of registration (UC-36 BR-3). | 1. Complete the Parent registration process. <br> 2. Observe the time taken for the confirmation email to be sent (check mock email service/logs). | The confirmation email is sent to the provided email address within 2 minutes of successful registration completion. (Requires backend email service integration and timing check). | The confirmation email is sent to the provided email address within 2 minutes of successful registration completion. (Requires backend email service integration and timing check). | pass |
| **BR\_UC37\_BR1** | Monitor Child's Progress Real-time Update Rule | Progress reports must be updated in real-time (UC-37 BR-1). | 1. Log in as a Parent and navigate to the dashboard monitoring a child's progress. <br> 2. As the child Student account, complete an activity that affects their progress metrics (e.g., finish a quiz). <br> 3. Observe the Parent's dashboard immediately after the child completes the activity. | The child's progress data (e.g., scores, completion percentages) is updated on the Parent's monitoring dashboard without requiring the parent to refresh the page. (Requires real-time communication). | The child's progress data (e.g., scores, completion percentages) is updated on the Parent's monitoring dashboard without requiring the parent to refresh the page. (Requires real-time communication). | pass |
| **BR\_UC38\_BR1** | Set Learning Goals Target Date Rule | Goals must have a target completion date (UC-38 BR-1). | 1. Log in as a Parent or Student (whoever sets goals). <br> 2. Go to the feature to set a learning goal. <br> 3. Attempt to save the goal without specifying a target completion date. | The attempt to save the goal fails. The UI displays a validation error indicating that a target completion date is a mandatory field. | The attempt to save the goal fails. The UI displays a validation error indicating that a target completion date is a mandatory field. | pass |
| **BR\_UC39\_BR2** | Apple Auth Direct Login Rule (Linked Account) | Users who have previously linked their Apple account do not need to grant permission again (UC-39 BR-2). | 1. Simulate a user who has previously linked their Apple account clicking "Login with Apple". | The user is logged in directly without being prompted to grant permissions via the Apple authorization dialog. | The user is logged in directly without being prompted to grant permissions via the Apple authorization dialog. | pass |
| **BR\_UC39\_BR3** | Apple Auth Privacy/Security Compliance Rule | Apple authentication must comply with Apple's privacy and security standards, including optional email hiding (UC-39 BR-3). | 1. On a new device/session, use "Login with Apple". <br> 2. Choose the option to hide email if presented. <br> 3. Complete the login. <br> 4. Verify the system receives and handles the private relay email. | The Apple authentication flow respects the user's privacy choice, specifically allowing email hiding. The system correctly processes the sign-in request using the private relay email address provided by Apple. (Requires specific setup for Apple's Sign In with Apple service). | The Apple authentication flow respects the user's privacy choice, specifically allowing email hiding. The system correctly processes the sign-in request using the private relay email address provided by Apple. (Requires specific setup for Apple's Sign In with Apple service). | pass |

## **Integration Testing:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | Test Description | Steps | Expected Result | Actual Result | Pass/Fail |
| **IT\_Auth\_001** | Registration Flow (UI -> Backend API -> DB -> Backend API -> UI) - Success | 1. On Sign Up screen, enter valid, unique Full Name, Username/Email, Password. <br> 2. Click "Signup". | The frontend correctly formats and sends a POST request to the real backend /register endpoint. The backend receives the request, validates (including BR\_UC1\_BR1, BR\_UC1\_BR2, BR\_BE\_Auth\_003 equivalent), saves to the DB, and returns a 201/200 success response. The frontend receives success and navigates to the next screen (e.g., Home or Login). | The frontend correctly formats and sends a POST request to the real backend /register endpoint. The backend receives the request, validates (including BR\_UC1\_BR1, BR\_UC1\_BR2, BR\_BE\_Auth\_003 equivalent), saves to the DB, and returns a 201/200 success response. The frontend receives success and navigates to the next screen (e.g., Home or Login). | pass |
| **IT\_Auth\_002** | Registration Flow (UI -> Backend API -> DB -> Backend API -> UI) - Duplicate Email Error | 1. On Sign Up screen, enter data including an email already in DB. <br> 2. Click "Signup". | The frontend sends the request to the real backend. The backend receives it, checks DB, detects duplicate email (BR\_UC1\_BR2), and returns an error response (e.g., 409 Conflict, 400 Bad Request) with details indicating the email conflict. The UI receives this specific error and displays it to the user. | The frontend sends the request to the real backend. The backend receives it, checks DB, detects duplicate email (BR\_UC1\_BR2), and returns an error response (e.g., 409 Conflict, 400 Bad Request) with details indicating the email conflict. The UI receives this specific error and displays it to the user. | pass |
| **IT\_Auth\_003** | Registration Flow (UI -> Backend API -> Backend API -> UI) - Invalid Data Error | 1. On Sign Up screen, enter data with invalid email format (e.g., " test@ "). <br> 2. Click "Signup". | The frontend sends the request to the real backend. The backend receives it, performs validation (if not done client-side), including email format checks, and returns a 400 Bad Request response with validation error details. The UI receives this and displays the validation error (related to BR\_UC31\_BR1 format requirement if applicable server-side). | The frontend sends the request to the real backend. The backend receives it, performs validation (if not done client-side), including email format checks, and returns a 400 Bad Request response with validation error details. The UI receives this and displays the validation error (related to BR\_UC31\_BR1 format requirement if applicable server-side). | pass |
| **IT\_Auth\_004** | Login Flow (UI -> Backend API -> DB -> Backend API -> UI) - Success | 1. On Login screen, enter valid credentials for an existing user. <br> 2. Click "Sign In". | The frontend sends a POST request to the real backend /login endpoint. The backend receives, verifies credentials against DB (BR\_UC2\_BR1), potentially generates a token, and returns a 200 OK response (with user data/token). The frontend receives success, stores user/session data locally, and navigates to the Home screen. | The frontend sends a POST request to the real backend /login endpoint. The backend receives, verifies credentials against DB (BR\_UC2\_BR1), potentially generates a token, and returns a 200 OK response (with user data/token). The frontend receives success, stores user/session data locally, and navigates to the Home screen. | pass |
| **IT\_Auth\_005** | Login Flow (UI -> Backend API -> Backend API -> UI) - Invalid Credentials Error | 1. On Login screen, enter valid Username/Email but incorrect Password. <br> 2. Click "Sign In". | The frontend sends the request to the real backend /login. The backend fails authentication (BR\_UC2\_BR1) and returns an error response (e.g., 401 Unauthorized, 400 Bad Request). The UI receives this and displays the invalid credentials message without navigating away from the login screen. | The frontend sends the request to the real backend /login. The backend fails authentication (BR\_UC2\_BR1) and returns an error response (e.g., 401 Unauthorized, 400 Bad Request). The UI receives this and displays the invalid credentials message without navigating away from the login screen. | pass |
| **IT\_Auth\_006** | Login Flow (UI -> Backend API -> DB -> Backend API -> UI) - User Not Found Error | 1. On Login screen, enter credentials for a non-existent user. <br> 2. Click "Sign In". | The frontend sends the request to the real backend /login. The backend queries the DB, fails to find the user (BR\_UC2\_BR1), and returns an error response (e.g., 404 Not Found, or 401 Unauthorized). The UI receives this and displays an error indicating user not found or invalid credentials. | The frontend sends the request to the real backend /login. The backend queries the DB, fails to find the user (BR\_UC2\_BR1), and returns an error response (e.g., 404 Not Found, or 401 Unauthorized). The UI receives this and displays an error indicating user not found or invalid credentials. | pass |
| **IT\_Auth\_007** | Logout Flow (UI -> Backend API - if applicable -> UI) | On Menu screen, click "Logout" (FT\_Menu\_169). | The frontend sends a request (e.g., POST to /logout) to the real backend if a server-side logout is needed to invalidate session/token. The backend processes this and returns a success response. The UI receives this, clears local user/session state, and navigates back to the Welcome/Login screen. | The frontend sends a request (e.g., POST to /logout) to the real backend if a server-side logout is needed to invalidate session/token. The backend processes this and returns a success response. The UI receives this, clears local user/session state, and navigates back to the Welcome/Login screen. | pass |
| **IT\_Auth\_008** | Forgot Password Flow (UI -> Backend API -> Email Service -> UI) | 1. On Login screen, click "Forgot Password?". <br> 2. On the Forgot Password screen, enter a registered email address. <br> 3. Submit the request. | The frontend sends a request to the real backend (e.g., /forgot-password). The backend receives the email, verifies it exists (BR\_UC2\_BR2), initiates the reset process, and interacts with the Email Service to send the reset email (verify via mock/real Email Service logs). The UI displays a success message indicating the email has been sent. | The frontend sends a request to the real backend (e.g., /forgot-password). The backend receives the email, verifies it exists (BR\_UC2\_BR2), initiates the reset process, and interacts with the Email Service to send the reset email (verify via mock/real Email Service logs). The UI displays a success message indicating the email has been sent. | pass |
| **IT\_Auth\_009** | Forgot Password Flow (UI -> Backend API -> UI) - alloysunregistered Email Error | 1. On Login screen, click "Forgot Password?". <br> 2. On the Forgot Password screen, enter an unregistered email address. <br> 3. Submit the request. | The frontend sends the request to the real backend. The backend receives the email, checks registration, determines it's unregistered (violating BR\_UC2\_BR2 implicitly), and returns an error. The UI receives this and displays an error message (e.g., "Email not found"). | The frontend sends the request to the real backend. The backend receives the email, checks registration, determines it's unregistered (violating BR\_UC2\_BR2 implicitly), and returns an error. The UI receives this and displays an error message (e.g., "Email not found"). | pass |
| **IT\_Auth\_010** | Social Login Flow (UI -> Auth Provider -> Backend API -> DB -> Backend API -> UI) - New User | 1. On Login/Signup, click "Login with Google" or "Login with Apple". <br> 2. Complete Auth Provider flow for a user not yet registered in the app. <br> 3. Grant permissions. | The system Auth Provider dialog/flow appears and is completed. The Frontend receives the Auth Provider token/info. The Frontend sends this to the real backend (e.g., /auth/google/callback). The backend verifies the token, checks if the user exists (BR\_UC3\_BR1/BR\_UC39\_BR1), registers a new user if not, links the social account, and returns success/token. The UI receives success and navigates to Home. | The system Auth Provider dialog/flow appears and is completed. The Frontend receives the Auth Provider token/info. The Frontend sends this to the real backend (e.g., /auth/google/callback). The backend verifies the token, checks if the user exists (BR\_UC3\_BR1/BR\_UC39\_BR1), registers a new user if not, links the social account, and returns success/token. The UI receives success and navigates to Home. | pass |
| **IT\_Auth\_011** | Social Login Flow (UI -> Auth Provider -> Backend API -> UI) - Existing Linked User | 1. On Login/Signup, click "Login with Google" or "Login with Apple". <br> 2. Complete Auth Provider flow for a user already linked to the app. | The system Auth Provider dialog/flow appears and is completed (may skip permission step due to BR\_UC3\_BR2/BR\_UC39\_BR2). The Frontend receives the token/info. The Frontend sends this to the real backend. The backend verifies the token, finds the existing linked user, and returns success/token. The UI receives success and navigates to Home. | The system Auth Provider dialog/flow appears and is completed (may skip permission step due to BR\_UC3\_BR2/BR\_UC39\_BR2). The Frontend receives the token/info. The Frontend sends this to the real backend. The backend verifies the token, finds the existing linked user, and returns success/token. The UI receives success and navigates to Home. | pass |
| **IT\_Auth\_012** | Delete Account Flow (UI -> Backend API -> DB) | 1. On Menu screen, click "Delete Account" (Implied Menu item). <br> 2. Confirm deletion if prompted. | The frontend triggers a request to the real backend /user/delete endpoint. The backend receives the request, performs deletion logic (including adhering to BR\_UC6\_BR1 about recovery), and responds with success. The UI receives success and navigates back to the Welcome/Login screen. (Verification of deletion in DB/logs is needed). | The frontend triggers a request to the real backend /user/delete endpoint. The backend receives the request, performs deletion logic (including adhering to BR\_UC6\_BR1 about recovery), and responds with success. The UI receives success and navigates back to the Welcome/Login screen. (Verification of deletion in DB/logs is needed). | pass |
|  |  |  |  |  | pass |
| **IT\_Sim\_Data\_013** | Home Screen Data Loading (UI loads & Renders Dummy Data) | 1. Log in successfully. <br> 2. The Home screen loads. | The Home screen components load and display dummy data for user details, scores/stats (BR\_UC21\_BR1, BR\_UC23\_BR1 display), dummy subject data (BR\_FE\_Home\_051 content), and dummy promotion data (BR\_FE\_Home\_049 content). The UI renders these sections correctly using the internal dummy data. | The Home screen components load and display dummy data for user details, scores/stats (BR\_UC21\_BR1, BR\_UC23\_BR1 display), dummy subject data (BR\_FE\_Home\_051 content), and dummy promotion data (BR\_FE\_Home\_049 content). The UI renders these sections correctly using the internal dummy data. | pass |
| **IT\_Sim\_Data\_014** | Overall Score Data Loading (UI loads & Renders Dummy Data) | 1. Navigate to the Overall Score screen (FT\_OverallScore\_134). | The Overall Score screen UI components load and display dummy data for overall score/grade (BR\_UC26\_BR1 display), subject breakdowns, and areas to improve (BR\_UC26\_BR2 display). The UI renders charts, lists, and metrics using this dummy data. | The Overall Score screen UI components load and display dummy data for overall score/grade (BR\_UC26\_BR1 display), subject breakdowns, and areas to improve (BR\_UC26\_BR2 display). The UI renders charts, lists, and metrics using this dummy data. | pass |
| **IT\_Sim\_Data\_015** | Subject Performance Data Loading (UI loads & Renders Dummy Data) | 1. Navigate to a Subject Performance screen (FT\_OverallScore\_140 or FT\_Home\_052). | The Subject Performance screen UI components load and display dummy data for subject-specific metrics (BR\_UC27\_BR1 display), chapter/topic performance (BR\_UC27\_BR2 display), and recommendations. The UI renders progress metrics, charts, and expandable lists using this dummy data. | The Subject Performance screen UI components load and display dummy data for subject-specific metrics (BR\_UC27\_BR1 display), chapter/topic performance (BR\_UC27\_BR2 display), and recommendations. The UI renders progress metrics, charts, and expandable lists using this dummy data. | pass |
| **IT\_Sim\_Data\_016** | Leaderboard Data Loading (UI loads & Renders Dummy Data) | 1. Navigate to the Leaderboard screen (FT\_Leaderboard\_152). | The Leaderboard screen UI components load and display dummy data for weekly and monthly rankings (BR\_FE\_Leaderboard\_029 display). The UI renders the podium and list based on this dummy data. Switching tabs updates the display to the corresponding dummy monthly data. | The Leaderboard screen UI components load and display dummy data for weekly and monthly rankings (BR\_FE\_Leaderboard\_029 display). The UI renders the podium and list based on this dummy data. Switching tabs updates the display to the corresponding dummy monthly data. | pass |
| **IT\_Sim\_Data\_017** | My Classes Data Loading (UI loads & Renders Dummy Data) | 1. Navigate to the My Classes screen (FT\_Classes\_124). | The My Classes screen UI components load and display a dummy list of classes. The UI renders this list of class cards using the internal dummy data (simulating BR\_UC16\_BR1 filtering). | The My Classes screen UI components load and display a dummy list of classes. The UI renders this list of class cards using the internal dummy data (simulating BR\_UC16\_BR1 filtering). | pass |
| **IT\_Sim\_Data\_018** | Class Details Data Loading (UI loads & Renders Dummy Data) | 1. Navigate to a Class Details screen (FT\_Classes\_126). | The Class Details screen UI components load and display dummy data for class details, materials, assignments, and forum data (simulating BR\_UC17\_BR1, BR\_UC18\_BR1, BR\_UC20\_BR1 filtering/data). The UI renders the sections and action buttons based on this dummy data. | The Class Details screen UI components load and display dummy data for class details, materials, assignments, and forum data (simulating BR\_UC17\_BR1, BR\_UC18\_BR1, BR\_UC20\_BR1 filtering/data). The UI renders the sections and action buttons based on this dummy data. | pass |
| **IT\_Sim\_Data\_019** | Battle History Data Loading (UI loads & Renders Dummy Data) | 1. Navigate to the Battle Lobby screen (FT\_Extras\_065). | The Battle Lobby screen UI components load and display dummy data for user info, banner, and a dummy list of battle history items (BR\_UC13\_BR1 display). The UI renders these sections using internal dummy data. | The Battle Lobby screen UI components load and display dummy data for user info, banner, and a dummy list of battle history items (BR\_UC13\_BR1 display). The UI renders these sections using internal dummy data. | pass |
| **IT\_Sim\_Data\_020** | Quiz Report Card Data Loading (UI loads & Renders Dummy Data) | 1. Complete a simulated battle/quiz (FT\_BrainBattle\_102, FT\_BattleResult\_107) OR Click a history item (FT\_BattleLobby\_074). | The Quiz Report Card screen UI components load and display dummy data simulating a quiz report. This includes dummy results, performance metrics (BR\_UC14\_BR1 display), question analysis (BR\_FE\_ReportCard\_016 display), and comparison data (BR\_FE\_ReportCard\_018 display). The UI renders the full report using this dummy data. | The Quiz Report Card screen UI components load and display dummy data simulating a quiz report. This includes dummy results, performance metrics (BR\_UC14\_BR1 display), question analysis (BR\_FE\_ReportCard\_016 display), and comparison data (BR\_FE\_ReportCard\_018 display). The UI renders the full report using this dummy data. | pass |
| **IT\_Sim\_Data\_021** | Notifications Data Loading (UI loads & Renders Dummy Data) | 1. Navigate to the Notifications screen (FT\_Home\_055). | The Notifications screen UI components load and display a dummy list of notifications. The UI renders the list of notification items using this internal dummy data. | The Notifications screen UI components load and display a dummy list of notifications. The UI renders the list of notification items using this internal dummy data. | pass |
| **IT\_Sim\_Interaction\_022** | Battle Initiation Flow (UI Trigger -> Navigation/Simulated Start) | 1. On Battle Lobby, click "Start the Battle" (FT\_BattleLobby\_072). <br> 2. On Select Subject, click a subject (FT\_SelectSubjectBattle\_081). <br> 3. On Battle Mode dialog, click "Whole Book" (FT\_BattleModeDialog\_085). <br> 4. On Invite Friends dialog, click "Random Match" (FT\_InviteFriendsDialog\_089). | The sequence of UI interactions (button clicks) triggers navigation between screens (Battle Lobby -> Select Subject) and display of modals (Battle Mode, Invite Friends). The final button click ("Random Match") on the last modal triggers navigation to the Brain Battle screen, initiating the simulated quiz flow. No actual backend battle setup is involved, only UI progression. | The sequence of UI interactions (button clicks) triggers navigation between screens (Battle Lobby -> Select Subject) and display of modals (Battle Mode, Invite Friends). The final button click ("Random Match") on the last modal triggers navigation to the Brain Battle screen, initiating the simulated quiz flow. No actual backend battle setup is involved, only UI progression. | pass |
| **IT\_Sim\_Interaction\_023** | Chapter Battle Initiation Flow (UI Trigger -> Navigation/Simulated Start w/ Context) | 1. On Battle Lobby, click "Start the Battle" (FT\_BattleLobby\_072). <br> 2. On Select Subject, click a subject (FT\_SelectSubjectBattle\_081). <br> 3. On Battle Mode dialog, click "By Chapter" (FT\_BattleModeDialog\_084). <br> 4. On Select Chapters, select chapters (FT\_SelectChapters\_092). <br> 5. Click "Start Match" (FT\_SelectChapters\_095). | The UI interaction sequence triggers navigation and modals. Selecting chapters updates UI state (IT\_UI\_State\_027). Clicking "Start Match" (enabled by BR\_FE\_Battle\_014 UI rule) triggers navigation to the Brain Battle screen, simulating a battle using questions relevant to the selected dummy chapters. No actual backend battle setup with selected chapters is involved. | The UI interaction sequence triggers navigation and modals. Selecting chapters updates UI state (IT\_UI\_State\_027). Clicking "Start Match" (enabled by BR\_FE\_Battle\_014 UI rule) triggers navigation to the Brain Battle screen, simulating a battle using questions relevant to the selected dummy chapters. No actual backend battle setup with selected chapters is involved. | pass |
| **IT\_Sim\_Interaction\_024** | Practice Initiation Flow (UI Trigger -> Navigation w/ Context) | 1. Click "Practice Now" (FT\_OverallScore\_139) or "Practice This Chapter" (FT\_SubjectPerf\_149). <br> 2. Arrive at a Practice screen. | The click event on the button triggers navigation. The Navigation system passes dummy parameters identifying the specific area/chapter (related to BR\_UC26\_BR2 or BR\_UC27\_BR2). The Practice screen component uses this dummy context to potentially filter and display dummy practice questions. No actual backend data filtering/loading is involved. | The click event on the button triggers navigation. The Navigation system passes dummy parameters identifying the specific area/chapter (related to BR\_UC26\_BR2 or BR\_UC27\_BR2). The Practice screen component uses this dummy context to potentially filter and display dummy practice questions. No actual backend data filtering/loading is involved. | pass |
| **IT\_Sim\_Interaction\_025** | Class Action Navigation Flow (UI Trigger -> Navigation) | 1. On Class Details screen, click an action button (e.g., "View Materials" FT\_ClassDetails\_129). <br> 2. Arrive at the corresponding screen (e.g., Class Materials). | Clicking the button triggers navigation to the target screen (Materials, Assignments, Forum). The Class ID context is passed via navigation. The target screen UI will likely load and display dummy data relevant to the Class ID. No backend action is performed upon clicking. | Clicking the button triggers navigation to the target screen (Materials, Assignments, Forum). The Class ID context is passed via navigation. The target screen UI will likely load and display dummy data relevant to the Class ID. No backend action is performed upon clicking. | pass |
| **IT\_Sim\_Interaction\_026** | Menu Item Navigation Flow (UI Trigger -> Navigation) | 1. On Menu screen, click a menu item (e.g., "Select Language" FT\_Menu\_161). <br> 2. Arrive at the corresponding screen/modal. | Clicking the menu item triggers navigation to the associated screen or modal (e.g., Account Management, Language Selection, Settings). The target UI will load and display dummy content. No backend action is performed upon clicking (except potentially Logout IT\_Auth\_007). | Clicking the menu item triggers navigation to the associated screen or modal (e.g., Account Management, Language Selection, Settings). The target UI will load and display dummy content. No backend action is performed upon clicking (except potentially Logout IT\_Auth\_007). | pass |
|  |  |  |  |  | pass |
| **IT\_Navigation\_027** | Navigation System - Stack Management (Login clears history) | 1. Start at Login. <br> 2. Enter valid credentials and click "Sign In" (IT\_Auth\_004). <br> 3. User navigates to Home. <br> 4. Use system back action/button. | After successful login and navigation to Home via the real backend call, the Navigation system should remove the Login screen from the stack. Using the system back action on Home should exit the app or move it to the background. | After successful login and navigation to Home via the real backend call, the Navigation system should remove the Login screen from the stack. Using the system back action on Home should exit the app or move it to the background. | pass |
| **IT\_Navigation\_028** | Navigation System - Bottom Nav Switching | 1. On Home, tap "Class" in bottom nav (FT\_BottomNav\_059). <br> 2. On Class screen, tap "Home" in bottom nav (FT\_BottomNav\_061). | Tapping a bottom nav item changes its visual state to active (BR\_FE\_BottomNav\_011 UI rule) and triggers navigation to the corresponding root screen (Home, Class). The screens are loaded/displayed using dummy data. Repeated taps on the active item stay on the same screen. | Tapping a bottom nav item changes its visual state to active (BR\_FE\_BottomNav\_011 UI rule) and triggers navigation to the corresponding root screen (Home, Class). The screens are loaded/displayed using dummy data. Repeated taps on the active item stay on the same screen. | pass |
| **IT\_Navigation\_029** | Navigation System - Header Back Button Consistency | 1. Navigate sequence Home -> Extras -> Battle -> Select Subject -> Select Chapters. <br> 2. Click the header back button repeatedly. | Clicking the header back button (present on most screens) consistently navigates back to the immediately preceding screen in the navigation stack (BR\_FE\_General\_031 UI rule). This tests the Navigation System's stack management on the frontend. | Clicking the header back button (present on most screens) consistently navigates back to the immediately preceding screen in the navigation stack (BR\_FE\_General\_031 UI rule). This tests the Navigation System's stack management on the frontend. | pass |
| **IT\_UI\_State\_030** | State Management: Chapter Selection impacts Button State (Select Chapters) | 1. On Select Chapters screen, observe the "Start Match" button's initial disabled state. <br> 2. Click a chapter checkbox (FT\_SelectChapters\_092). | The click event from the ChapterItem component is handled by the parent screen's state logic. This logic updates the internal state tracking selected chapters (increment count). The "Start Match" button component reads this internal state (selected count > 0) and updates its own state to enabled, following the BR\_FE\_Battle\_014 UI rule. This is purely frontend state management. | The click event from the ChapterItem component is handled by the parent screen's state logic. This logic updates the internal state tracking selected chapters (increment count). The "Start Match" button component reads this internal state (selected count > 0) and updates its own state to enabled, following the BR\_FE\_Battle\_014 UI rule. This is purely frontend state management. | pass |
| **IT\_UI\_State\_031** | State Management: Tab Selection impacts Content Display (Overall Score) | 1. On Overall Score screen, observe the content under the "Subject Breakdown" tab. <br> 2. Click the "Areas to Improve" tab (FT\_OverallScore\_137). | Clicking the tab updates the screen component's internal state for the active tab (BR\_FE\_OverallScore\_021 UI rule). Based on this state, the screen component logic renders the correct content section ("Areas to Improve" components/dummy data) and hides the previous content ("Subject Breakdown" components/dummy data). This is purely frontend state management and rendering. | Clicking the tab updates the screen component's internal state for the active tab (BR\_FE\_OverallScore\_021 UI rule). Based on this state, the screen component logic renders the correct content section ("Areas to Improve" components/dummy data) and hides the previous content ("Subject Breakdown" components/dummy data). This is purely frontend state management and rendering. | pass |
| **IT\_UI\_Display\_032** | Data Display Integration: List Component renders Item Components with dummy data | 1. Navigate to a screen with a list populated from dummy data (e.g., Battle Lobby, My Classes). <br> 2. Observe the list items displayed. | The screen component has access to an array of dummy data. The List component iterates through this array and renders an Item component for each entry, passing the relevant dummy data fields to each item component for display (e.g., BR\_UC13\_BR1 display for Battle History, BR\_UC16\_BR1 display for My Classes). This confirms the List and Item components integrate correctly with a local data source. | The screen component has access to an array of dummy data. The List component iterates through this array and renders an Item component for each entry, passing the relevant dummy data fields to each item component for display (e.g., BR\_UC13\_BR1 display for Battle History, BR\_UC16\_BR1 display for My Classes). This confirms the List and Item components integrate correctly with a local data source. | pass |
| **IT\_UI\_Display\_033** | Data Display Integration: Item Component state based on dummy data (Report Card Question) | 1. Navigate to a Quiz Report Card screen (IT\_Sim\_Data\_020). <br> 2. Observe the Question Analysis list items. | The Report Card screen component receives dummy data simulating detailed question results. The List component renders ReportCardQuestionItem components. Each item receives dummy data including user answer, correct answer, and correctness flag. The item component uses this dummy data to display the correct info and the appropriate visual indicator (X for incorrect, related to BR\_FE\_ReportCard\_016 UI rule). | The Report Card screen component receives dummy data simulating detailed question results. The List component renders ReportCardQuestionItem components. Each item receives dummy data including user answer, correct answer, and correctness flag. The item component uses this dummy data to display the correct info and the appropriate visual indicator (X for incorrect, related to BR\_FE\_ReportCard\_016 UI rule). | pass |
| **IT\_UI\_Interaction\_034** | Modal Interaction Integration (Widget Click -> Modal Display -> Modal Close) | 1. On Home screen, click the Score widget (FT\_Home\_045). <br> 2. The Score & Goals modal appears (BR\_FE\_Home\_009 UI rule). <br> 3. Click the 'X' button on the modal (FT\_Modal\_044). | Clicking the widget triggers the Home screen logic to change a state variable controlling modal visibility. The Modal component reads this state and becomes visible over the Home screen. Clicking the 'X' button on the Modal component triggers its internal logic/a callback to the parent to update the state variable, causing the modal to become hidden. This is purely frontend component interaction. | Clicking the widget triggers the Home screen logic to change a state variable controlling modal visibility. The Modal component reads this state and becomes visible over the Home screen. Clicking the 'X' button on the Modal component triggers its internal logic/a callback to the parent to update the state variable, causing the modal to become hidden. This is purely frontend component interaction. | pass |