Super Mario

Mobile Application

Submitted By:

Hassan Rajput - 22SW108

Dayaan Mughal - 22SW150

Submitted To:

Ma’am Mariam Memon

**1. Real World Problem Identification**

**Problem Statement**

Need for an interactive mobile game that:

* Provides entertainment
* Offers a simple, engaging mechanic
* Tracks player progress over time
* Runs efficiently on mobile devices
* Offers Nostalgic experience

**Key Requirements**

1. User engagement: simple controls and clear objectives
2. Progress tracking: persistent high scores across sessions
3. Performance: smooth animations and responsive controls
4. Accessibility: easy to learn, suitable for various age groups
5. Cross-platform support: works on Android, iOS, and other platforms

**Target Audience**

* Casual mobile gamers
* Players seeking quick sessions
* Ages 8+

**2. Proposed Solution**

**Solution Overview**

Developed a Super Mario-themed collection game using Flutter, where players control Mario to collect mushrooms within a 60-second timer.

**Core Features**

1. Character control

* Left/right movement with running animation
* Jumping with parabolic physics

1. Game mechanics

* Collect mushrooms to increase score
* 60-second timer
* Random mushroom spawns
* Character growth with each collection

1. User interface

* On-screen directional controls
* Real-time stats (score, time, high score)
* Game-style typography (Press Start 2P)
* Landscape orientation

1. Progress tracking

* High score with date stamp

**Technical Architecture**

* Framework: Flutter (Dart)
* Architecture: Stateful widget with State management
* Physics: Custom jump mechanics using parabolic equations
* Animation: Timer-based position updates with smooth transitions

**3. Data Storage**

Selected Database: Firebase Firestore  
Firestore was chosen as the scalable cloud NoSQL document database.

Justification

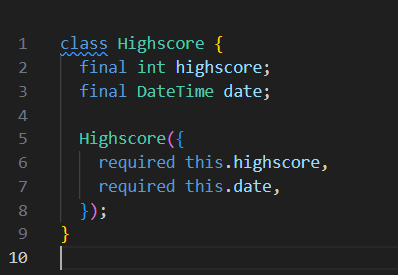
1. Real-time Synchronization  
   • Instantly updates data across all clients  
   • Enables dynamic and interactive gameplay updates  
   • Ideal for multiplayer or live score tracking
2. Scalability  
   • Handles large datasets efficiently  
   • Automatically scales with user base and data load  
   • Supports global distribution
3. Cloud-based Storage  
   • Data stored securely in the cloud  
   • Accessible from any device  
   • No manual backup required
4. Integration with Firebase Ecosystem  
   • Seamless setup with Firebase Authentication, Analytics, and Hosting  
   • Works smoothly with FlutterFire plugins  
   • Simplifies backend management
5. Offline Persistence  
   • Automatically caches data locally  
   • Ensures smooth gameplay even without internet  
   • Syncs changes once connection is restored
6. Security  
   • Uses Firebase Security Rules for fine-grained access control  
   • Supports authentication-based data protection  
   • Managed by Google Cloud infrastructure
7. Ease of Use  
   • Simple read/write API for developers  
   • Supports real-time listeners and async operations  
   • Comprehensive documentation and community support
8. Learning Purposes  
   • Enhances understanding of cloud databases  
   • Provides experience in real-world app integration  
   • Builds knowledge of scalable and modern backend solutions

**Top of Form**

**Bottom of Form**

**Implementation Details**

**Data Model:**



**User Interface:**

A screen shot of a video game

AI-generated content may be incorrect.

**Storage Operations:**

* **Write: addDate method**

A screen shot of a computer program

AI-generated content may be incorrect.

* **Read: fetchData method**

A screen shot of a computer program

AI-generated content may be incorrect.

**4. APIs/Packages/Plug-ins**

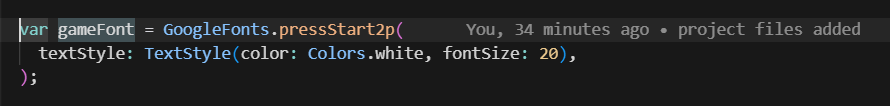
**4.1 Google Fonts (google\_fonts: ^6.3.2)**

**Purpose:** Game style Typography

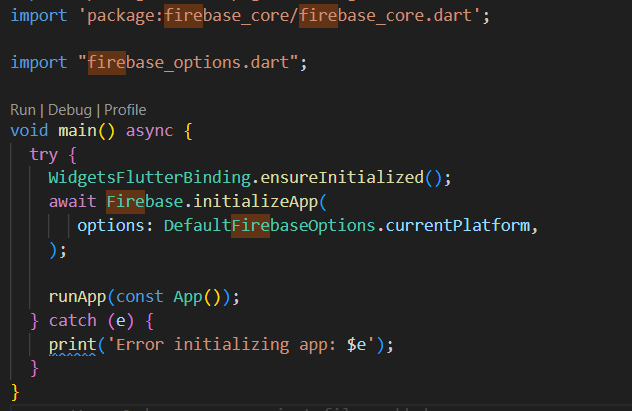
**Justification:**

* Press Start 2P font matches classic game aesthetics

**Usage:**

****

**4.2 firebase\_core: ^4.2.0** Purpose: Core Firebase integration for Flutter apps  
 **Justification:** • Initializes and configures Firebase in the Flutter environment  
 • Provides connection between Flutter and Firebase services  
 • Essential for using other Firebase plugins like Firestore or Auth  
 **Usage:** • App-level Firebase initialization  
 • Ensures proper setup before using Firestore or any Firebase service



**4.3 cloud\_firestore: ^6.0.3** Purpose: Cloud-hosted NoSQL database for real-time data storage and synchronization

**Justification:** • Scalable and flexible document-based database  
 • Enables real-time updates across all connected clients  
 • Secure data handling with Firebase Authentication and Security Rules  
 **Usage:** • Storing and retrieving high scores in the cloud  
 • Syncing player progress across devices  
 • Maintaining high-score history

**5. Issues and Bugs Encountered and Resolved During Development**

**Issue 1: Firebase Initialization Errors**

A screen shot of a computer

AI-generated content may be incorrect.

**Solution:**

*Created the firebase app with FlutterFire dependency*

**Issue 2: cloud firestore permission denied**

**A black screen with white text

AI-generated content may be incorrect.**

**Solution:**

Logged in to the firebase using flutterfire

**Issue 3: Flutterfire Cli error when login in**

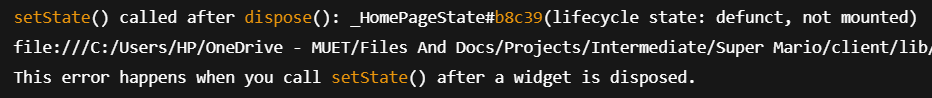
A screen shot of a computer

AI-generated content may be incorrect.

**Solution:**

Install flutter fire new version

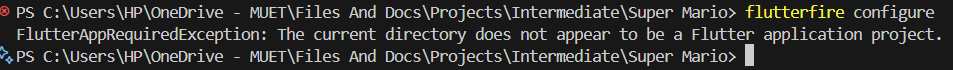
**Issue 4: Timer Not Canceling on Game End**



**Solution:**

Created a dispose method to cancel the timer method

**Issue 5: Multiple Simultaneous Button Presses**



**Solution** did cd to next directory