**Updated Comprehensive Plan for Book Reading Tracker Plugin**

Based on my research, here's a carefully considered plan addressing all your requirements, including the critical challenge with Goodreads API being deprecated since December 2020 [Goodreads](https://www.goodreads.com/topic/show/21788520-api-deprecation)[Medium](https://debugger.medium.com/goodreads-is-retiring-its-current-api-and-book-loving-developers-arent-happy-11ed764dd95).

**Critical Issue: Book Data Integration Strategy**

Since Goodreads no longer issues new developer keys for their public API [Goodreads Developers - API Deprecation Showing 1-50 of 64](https://www.goodreads.com/topic/show/21788520-api-deprecation), we need alternative approaches:

**Recommended Book Data Sources (Priority Order)**

1. **Google Books API** (PRIMARY)
   * Free, comprehensive, well-maintained
   * 1000+ requests per day free quota
   * ISBN, title, and author search
   * Cover images and descriptions
2. **Open Library API** (SECONDARY)
   * Free and open source alternative [Best Goodreads Alternatives: Top Social Networks in 2025 | AlternativeTo](https://alternativeto.net/software/goodreads/)
   * Complete API access
   * Community-driven metadata
   * Works well for older/rare books
3. **Hardcover API** (OPTIONAL)
   * GraphQL API that's free-to-use and developer-friendly [Hardcover's Book API: A great Goodreads alternative](https://www.emgoto.com/hardcover-book-api/)
   * Modern alternative specifically designed to replace Goodreads
   * Good for social features integration
4. **Hybrid Approach with User Import**
   * Allow CSV import from Goodreads export
   * StoryGraph import capability
   * Manual entry with ISBN lookup
   * Community-contributed book database within Moodle

**Architecture Overview - Modular Design**

┌─────────────────────────────────────────────────────────────┐

│ MOODLE CORE SYSTEMS │

├─────────────────┬──────────────────┬────────────────────────┤

│ Activity │ Gamification │ User Management │

│ Completion │ (Badges/XP) │ (Groups/Roles) │

└─────────────────┴──────────────────┴────────────────────────┘

▲

│ Integration Layer

│

┌─────────────────────────────────────────────────────────────┐

│ BOOK READER PLUGIN (mod\_bookreader) │

├──────────────┬────────────────┬──────────────────────────────┤

│ Book Data │ Social/Review │ Gamification Engine │

│ Management │ Module │ (Points & Leaderboards) │

├──────────────┼────────────────┼──────────────────────────────┤

│ Charting │ API Gateway │ Activity Integration │

│ & Analytics │ (External) │ (Homework/Quiz Points) │

└──────────────┴────────────────┴──────────────────────────────┘

**Database Schema (Enhanced)**

sql

*-- Core book data with multiple source support*

CREATE TABLE mdl\_bookreader\_books (

id BIGINT AUTO\_INCREMENT PRIMARY KEY,

google\_books\_id VARCHAR(255),

open\_library\_id VARCHAR(255),

isbn\_10 VARCHAR(13),

isbn\_13 VARCHAR(17),

title VARCHAR(255) NOT NULL,

authors TEXT,

publisher VARCHAR(255),

published\_date DATE,

page\_count INT,

categories TEXT,

language\_code VARCHAR(10),

cover\_url TEXT,

description TEXT,

metadata JSON, *-- Flexible storage for API-specific data*

source VARCHAR(50), *-- 'google', 'openlibrary', 'manual', etc.*

INDEX idx\_isbn (isbn\_10, isbn\_13),

INDEX idx\_title (title),

FULLTEXT idx\_search (title, authors)

);

*-- Unified activity points system*

CREATE TABLE mdl\_bookreader\_points\_config (

id BIGINT AUTO\_INCREMENT PRIMARY KEY,

activity\_type VARCHAR(50), *-- 'book\_page', 'book\_complete', 'quiz', 'assignment'*

points\_formula TEXT, *-- JSON formula for calculation*

multipliers JSON, *-- Category/difficulty multipliers*

integration\_plugin VARCHAR(100) *-- NULL for books, 'mod\_quiz', 'mod\_assign', etc.*

);

*-- Comprehensive leaderboard with activity tracking*

CREATE TABLE mdl\_bookreader\_leaderboard\_entries (

id BIGINT AUTO\_INCREMENT PRIMARY KEY,

userid BIGINT,

groupid BIGINT,

period\_start DATETIME,

period\_end DATETIME,

book\_points INT DEFAULT 0,

activity\_points INT DEFAULT 0, *-- From homework/quizzes*

total\_points INT GENERATED ALWAYS AS (book\_points + activity\_points),

pages\_read INT DEFAULT 0,

books\_completed INT DEFAULT 0,

activities\_completed INT DEFAULT 0,

rank INT,

INDEX idx\_period\_group (groupid, period\_start, period\_end),

INDEX idx\_user\_period (userid, period\_start)

);

**Interactive Charting Implementation**

Since Moodle 3.2+ includes Chart.js with interactive, dynamic and responsive charts [Create interactive reports with new and dynamic graphs in Moodle 3.2 - Moodle](https://moodle.com/news/create-interactive-reports-new-dynamic-graphs-moodle-3-2/), we'll use a hybrid approach:

**Primary: Chart.js (Built-in)**

javascript

*// Using Moodle's native Chart API*

define(['core/chartjs'], function(Chart) {

return {

renderProgressChart: function(containerId, data) {

const chart = new Chart(document.getElementById(containerId), {

type: 'line',

data: {

datasets: [{

label: 'Pages Read',

data: data.daily,

borderColor: 'rgb(75, 192, 192)',

tension: 0.1

}]

},

options: {

responsive: true,

interaction: {

mode: 'index',

intersect: false

},

scales: {

x: {

type: 'time',

time: {

unit: 'day'

}

}

}

}

});

}

};

});

**Advanced: D3.js for Complex Visualizations**

For more complex interactive features like brushing, zooming, and dragging [D3.js](https://d3js.org/):

javascript

*// Interactive reading heatmap*

define(['jquery', 'd3'], function($, d3) {

return {

createReadingHeatmap: function(container, data) {

*// Interactive calendar heatmap showing reading intensity*

const svg = d3.select(container)

.append('svg')

.attr('width', 960)

.attr('height', 136);

*// Add zoom behavior*

const zoom = d3.zoom()

.scaleExtent([1, 8])

.on('zoom', zoomed);

svg.call(zoom);

*// ... implementation*

}

};

});

**Integration with Moodle's Gamification Ecosystem**

**1. Level Up XP Integration**

Since Level Up XP is the most popular gamification plugin with 27,000 sites [Moodle Plugins directory: Level Up XP - Gamification 🏅 | Moodle.org](https://moodle.org/plugins/block_xp), we'll integrate with it:

php

*// Hook into XP events*

function bookreader\_xp\_event\_observer($event) {

if ($event->component == 'mod\_bookreader') {

*// Award XP points for reading milestones*

\block\_xp\local\xp\event\_processor::process\_event($event);

}

}

**2. Activity Completion Integration**

Leverage Moodle's activity completion tracking [Moodle Plugins directory: Ranking block | Moodle.org](https://moodle.org/plugins/block_ranking) for unified points:

php

class activity\_points\_aggregator {

public function calculate\_total\_points($userid, $courseid, $period) {

$points = 0;

*// Book reading points*

$points += $this->get\_reading\_points($userid, $period);

*// Quiz points (if mod\_quiz installed)*

if (core\_plugin\_manager::instance()->get\_plugin\_info('mod\_quiz')) {

$points += $this->get\_quiz\_points($userid, $courseid, $period);

}

*// Assignment points*

$points += $this->get\_assignment\_points($userid, $courseid, $period);

*// Forum participation*

$points += $this->get\_forum\_points($userid, $courseid, $period);

return $points;

}

}

**Implementation Timeline (12-14 weeks)**

**Phase 1: Foundation & API Integration (Weeks 1-3)**

* Set up plugin structure with proper namespace
* Implement Google Books API integration
* Create fallback to Open Library API
* Build ISBN lookup service
* CSV import for Goodreads/StoryGraph data

**Phase 2: Core Reading Features (Weeks 4-5)**

* Reading progress tracking
* Book search with filters (title, author, ISBN)
* Personal library management
* Reading status updates
* Basic progress visualization with Chart.js

**Phase 3: Social Features (Weeks 6-7)**

* Review system with rich text editor
* Like/comment functionality
* Following system
* Activity feeds
* Email notifications for social interactions

**Phase 4: Advanced Gamification (Weeks 8-9)**

* Points calculation engine with configurable formulas
* Integration with Level Up XP (if installed)
* Activity completion points aggregation
* Monthly leaderboard with automatic reset
* Group-based competitions
* Achievement badges using Moodle's badge system

**Phase 5: Interactive Analytics (Weeks 10-11)**

* Interactive time-period selector (daily/weekly/monthly/yearly/all-time)
* Reading velocity charts
* Genre distribution pie charts
* Reading streak visualization
* Comparative analytics (user vs group average)
* D3.js implementation for advanced interactions

**Phase 6: Integration & Polish (Weeks 12-14)**

* Homework/quiz points integration
* Mobile responsive design
* Performance optimization with caching
* Accessibility (WCAG 2.1 AA)
* Documentation and help system
* Beta testing with real users

**Technical Implementation Details**

**API Service Layer**

php

namespace mod\_bookreader\external;

class book\_api\_service {

private $providers = [];

public function \_\_construct() {

$this->providers[] = new google\_books\_provider();

$this->providers[] = new open\_library\_provider();

if (get\_config('mod\_bookreader', 'enable\_hardcover')) {

$this->providers[] = new hardcover\_provider();

}

}

public function search($query, $type = 'all') {

$results = [];

foreach ($this->providers as $provider) {

try {

$results = array\_merge($results, $provider->search($query, $type));

} catch (\Exception $e) {

*// Log and continue with next provider*

debugging('Book API error: ' . $e->getMessage());

}

}

return $this->deduplicate\_results($results);

}

}

**Interactive Chart Components**

javascript

*// Time period selector with live updates*

class PeriodSelector {

constructor(chartInstance) {

this.chart = chartInstance;

this.periods = ['daily', 'weekly', 'monthly', 'yearly', 'all-time'];

this.init();

}

init() {

this.selector = document.createElement('div');

this.selector.className = 'chart-period-selector btn-group';

this.periods.forEach(period => {

const btn = this.createButton(period);

btn.addEventListener('click', () => this.updatePeriod(period));

this.selector.appendChild(btn);

});

}

async updatePeriod(period) {

const data = await this.fetchData(period);

this.chart.data = this.transformData(data, period);

this.chart.update('active');

}

}

**Risk Mitigation Strategies**

1. **API Dependency**: Implement robust caching and fallback mechanisms
2. **Performance**: Use Moodle's caching API for leaderboards
3. **Scalability**: Implement pagination and lazy loading
4. **Data Privacy**: Follow GDPR with Privacy API implementation
5. **Browser Compatibility**: Progressive enhancement approach

**Recommended Development Approach**

1. **Start with MVP**: Basic book tracking + simple leaderboard
2. **Iterative Enhancement**: Add features based on user feedback
3. **API Abstraction**: Design for easy addition of new book data sources
4. **Modular Architecture**: Each feature as a separate subplugin
5. **Test-Driven Development**: PHPUnit and Behat tests from day one

**Future Enhancements**

* AI-powered book recommendations
* Reading challenges and goals
* Book club functionality
* Integration with e-readers (Kindle, Kobo)
* Audio book tracking
* Speed reading metrics
* Vocabulary builder integration