D424 – Software Capstone

WGU Cloud Planner

Capstone Proposal Project Name: WGU Cloud Planner

Student Name: Andrew Davis Maloch

Table of Contents

Business Problem

The Customer

The primary customers are Western Governors University students pursuing degree programs through the university's competency-based education model. WGU serves over 180,000 students nationwide, with each student managing an average of 10-15 courses per academic year. Key stakeholders include students managing complex degree plans, academic mentors providing guidance, and the institution itself which benefits from improved student organization and progress tracking. The current IT infrastructure consists of individual mobile devices with isolated local storage, lacking centralized data synchronization capabilities.

Business Case

Currently, WGU students face significant challenges in maintaining consistent degree progress tracking across multiple devices. When students switch between mobile phones, tablets, or replace devices, they risk losing critical academic planning data or encountering version conflicts. This data fragmentation leads to inefficient manual synchronization efforts, potential loss of academic progress history, and reduced confidence in long-term planning. The absence of cloud persistence creates vulnerability to device failure and limits opportunities for collaborative planning with academic mentors.

The proposed cloud-enabled Degree Plan Tracker will solve these problems by providing automatic multi-device synchronization, secure cloud backup of academic data, and seamless transition between platforms. This centralized approach ensures students always have access to their current degree plan regardless of device used, while maintaining data integrity and academic history.

Fulfillment

The software application will fulfill student needs through a .NET MAUI mobile interface integrated with Google Firebase cloud services. The application will maintain all existing degree planning functionality while adding secure user authentication, real-time Firestore database synchronization, and cross-device data consistency. Basic functions will include academic term management, course progress tracking, assessment deadline monitoring, and instructor information storage - all synchronized instantly across all user devices.

The system will interface with Firebase Authentication for secure user access and Firebase Firestore for cloud data persistence. Users will access functions through an intuitive mobile interface with seamless offline-to-online transition capabilities. The outcome will be presented as a consistently updated academic dashboard accessible from any device, providing reliable progress tracking, automated backup protection, and elimination of manual synchronization efforts that currently burden students.

SDLC Methodology

The SDLC methodology selected for this project is an Adapted Agile approach with one-week sprints. This methodology is ideal for this project because it combines Agile's flexibility and iterative nature with the time constraints of a 25-day development timeline. While traditional Agile uses 2-4 week sprints, the compressed one-week approach maintains core Agile principles—iterative development, continuous testing, and adaptive planning—while ensuring timely delivery. This is particularly suitable since the project builds upon an existing codebase with well-defined requirements, but still requires flexibility for cloud integration challenges.

The Adapted Agile methodology will include three sequential sprints:

- Sprint 1: Authentication & Foundation
- Sprint 2: Data Migration & Sync
- Sprint 3: Testing & Deployment

Deliverables

Project Deliverables

- Sprint Backlogs: Detailed task lists for each one-week sprint, including authentication implementation, data migration tasks, and testing activities
- Project Timeline: 25-day schedule with specific milestones, dependencies, and resource allocation
- Test Plans: Comprehensive testing strategies for Firebase integration, including sync validation, authentication testing, and cross-device functionality

- Requirements Traceability Matrix: Mapping of capstone requirements to implemented features and documentation
- Risk Assessment: Identification of potential challenges in cloud migration and mitigation strategies

Product Deliverables

- Deployed Mobile Application: Production-ready .NET MAUI application published to Microsoft App Center for distribution
- Firebase Cloud Infrastructure: Fully configured Firebase project with Authentication, Firestore database, and security rules
- Source Code Repository: Complete codebase with Firebase integration, data migration layer, and synchronization logic
- User Documentation: Comprehensive guides covering authentication setup, multi-device synchronization, and feature usage
- Technical Architecture Documentation: Detailed overview of cloud integration patterns, data models, and system design decisions

Deployment Plan and Outcomes

Development Plan & Anticipated Outcomes

The development plan follows a structured three-sprint approach focused on incremental cloud integration, with each phase building toward the final deployed product:

Sprint 1: Foundation & Authentication (Days 1-6)

- *Development Focus:* Firebase project initialization, authentication service integration, user registration/login workflows
- *Anticipated Outcome*: Secure user management system enabling multi-user support and personalized data access

Sprint 2: Data Migration & Synchronization (Days 7-16)

- *Development Focus:* SQLite to Firestore data layer migration, real-time synchronization engine, offline capability implementation
- *Anticipated Outcome*: Seamless multi-device data consistency with robust conflict resolution and persistent cloud backup

Sprint 3: Testing & Deployment Preparation (Days 17-25)

- *Development Focus:* Comprehensive testing, performance optimization, and deployment packaging
- *Anticipated Outcome:* Production-ready application with validated cloud functionality and distribution-ready packaging

Deployment Strategy

The deployment will execute in the final development phase (Days 17-25) using a phased approach that ensures stability and validates all cloud components before user access:

Deployment Timeline & Validation:

- Days 17-19: Firebase Production Configuration
  - Security rules implementation and production environment setup
  - *Validation Checkpoint:* Security rules tested against simulated user access patterns
- Days 20-22: Application Packaging & Distribution
  - APK generation and distribution platform configuration (Microsoft App Center)
  - *Validation Checkpoint:* Installation and basic functionality testing on target devices
- Days 23-25: Final Deployment & User Provisioning
  - Production environment activation and user access enablement
  - *Verification Checkpoint:* End-to-end testing of authentication, data sync, and multi-device functionality

Anticipated Deployment Outcomes

Upon successful deployment, the software product will deliver:

1. Enterprise-Grade Availability: 24/7 cloud access to academic data across all user devices
2. Seamless User Transition: Existing local data automatically migrates to cloud storage upon first authentication
3. Scalable Infrastructure: Firebase backend supporting unlimited user growth within academic context
4. Professional Distribution: Application available through standardized mobile distribution channels

Implementation Roles & Dependencies

Primary Resource: Solo Developer (Project Lead)

- Manages deployment sequencing, cloud configuration, and final validation
- Coordinates testing across multiple physical devices and environments

Critical Deployment Dependencies:

- Firebase production configuration requires successful data migration testing (Day 16)
- Application packaging depends on resolution of all critical functionality issues
- Final deployment requires completion of capstone documentation deliverables

Deployment Sequence:

Firebase Production Setup → Application Packaging → Distribution Deployment → Final Validation

Project Timeline

Phase	Milestone/Task	Resources	Dependencies	Deliverable	Description	Dates w/ Duration
Foundation	Firebase Setup & Auth	Solo Developer	Task 1 Approval	Authentication System	Implement Firebase Auth, user registration/login flows	Oct 4 - Oct 9, 2025
Data Migration	SQLite to Firestore Migration	Solo Developer	Auth Completion	Cloud Data Layer	Migrate Term/Course/Assessment models to Firestore with real-time sync	Oct 10 - Oct 19, 2025
Testing & Deployment	Cloud Feature Validation	Solo Developer + Testing Devices	Data Migration Completion	Deployed Application	Comprehensive testing across devices, app deployment to distribution platform	Oct 20 - Oct 28, 2025
Documentation	Part 1: Requirements	Solo Developer	Project Approval	Software Requirements Spec	Formal documentation of project requirements and scope	Oct 4 - Oct 11, 2025
Documentation	Part 2: Design & Development	Solo Developer	Development Progress	Technical Documentation	System architecture, design decisions, development process	Oct 12 - Oct 21, 2025
Documentation	Part 3: QA & Deployment	Solo Developer	Testing Completion	Test & Deployment Docs	Test plans, deployment procedures, user guides	Oct 22 - Oct 26, 2025
Documentation	Part 4: Retrospective	Solo Developer	Project Completion	Final Presentation	Project retrospective, lessons learned, video presentation	Oct 27 - Oct 28, 2025

Environments and Costs

Programming Environment

The software product will be developed and deployed using the following technology stack:

- Development IDE: Visual Studio 2022 Community Edition with .NET MAUI workload
- Mobile Framework: .NET MAUI (Multi-platform App UI) for cross-platform mobile deployment
- Cloud Backend: Google Firebase platform (Authentication, Firestore Database)
- Development OS: Windows 11 with Android emulator and physical device testing
- Distribution Platform: Microsoft App Center for application deployment and distribution
- Version Control: Git with GitHub repository for source code management
- Documentation Tools: Microsoft Office Suite for documentation deliverables

Environment Costs

The total environment costs for this project are $0, utilizing free tiers and academic licensing:

- Visual Studio 2022 Community Edition: $0 (free for individual developers)
- .NET MAUI Framework: $0 (open-source, cross-platform framework)
- Google Firebase Spark Plan: $0 (free tier includes 10GB storage, 50K/day authentication requests)
- Microsoft App Center: $0 (free tier includes app distribution and basic analytics)
- GitHub Repository: $0 (free public/private repositories)

All services operate within their free tier limitations, which adequately support the project's scale and user base. The Firebase Spark Plan provides sufficient capacity for authentication and data storage needs, while Microsoft App Center offers robust application distribution without licensing costs.

Human Resource Requirements

Labor Allocation and Costs

The total development effort requires 144 hours distributed across planning, development, and documentation activities. All tasks will be executed by a single software engineer serving in multiple roles.

Human Resource Breakdown:

- Software Engineer/Architect: 88 hours (61%) - Firebase integration, data migration, authentication implementation
- QA Engineer: 16 hours (11%) - Testing, validation, and deployment verification
- Technical Writer: 36 hours (25%) - Documentation, requirements specification, project reporting
- Project Manager: 4 hours (3%) - Timeline management, milestone tracking, resource coordination

Total Labor Cost: $0 (academic project, no monetary compensation)

The human resource distribution reflects the project's technical complexity, with the majority of effort dedicated to cloud integration and data migration tasks. Documentation represents a significant portion (25%) of the total effort, aligning with capstone requirements for comprehensive project documentation. The solo developer model ensures consistent architectural decisions and efficient knowledge transfer throughout the project lifecycle.

Software Testing Approach

1. Validation and Verification Methods

The software product will undergo comprehensive testing to ensure it meets customer needs through multiple validation approaches:

Functional Validation Testing:

- Authentication Flow Testing: Verify user registration, login, password reset, and session management work seamlessly across devices
- Data Synchronization Testing: Confirm real-time sync of academic data (terms, courses, assessments) between multiple devices and cloud storage
- Offline Capability Testing: Validate application functionality during network outages with proper sync recovery when connectivity resumes
- Migration Integrity Testing: Ensure all existing SQLite data successfully migrates to Firestore without loss or corruption

User Experience Verification:

- Cross-Device Consistency Testing: Verify identical functionality and data display across different mobile devices and screen sizes

- Performance Benchmarking: Measure sync times, authentication speed, and application responsiveness against acceptable thresholds
- Error Handling Validation: Test graceful degradation during network failures, invalid inputs, and edge cases

Technical Implementation Testing:

- Firebase Security Rules Testing: Validate data access controls and user isolation in the cloud environment
- API Integration Testing: Verify proper communication between .NET MAUI client and Firebase services
- Data Model Integrity Testing: Confirm Firestore data structure maintains relational consistency for academic planning

All testing will be performed by the solo developer using a combination of physical Android devices and emulators to simulate real-world usage scenarios. The testing approach employs both automated unit tests for core functions and manual testing for user experience validation.

2. Test Results Analysis

Test results will be analyzed through both quantitative metrics and qualitative assessment to ensure comprehensive quality assurance:

Quantitative Analysis:

- Sync Performance Metrics: Measure and compare data synchronization times across devices, targeting sub-2-second sync completion for academic data updates
- Authentication Success Rates: Track login/registration success percentages, aiming for 99%+ success rate across test cycles
- Error Frequency Tracking: Monitor and categorize application errors, working toward zero critical errors in final testing phases
- Data Integrity Scores: Verify 100% data preservation during SQLite-to-Firestore migration and subsequent sync operations

Qualitative Assessment:

- User Workflow Validation: Confirm all academic planning workflows (term creation, course enrollment, assessment tracking) function identically to the local-only version while gaining cloud benefits
- Cross-Device Experience Consistency: Ensure seamless transition between devices without user-visible data conflicts or synchronization issues
- Error Message Clarity: Evaluate that all error states provide clear, actionable guidance to users

Acceptance Criteria Evaluation:

The test results will be measured against predefined acceptance criteria including:

- All existing degree planning features must maintain full functionality
- Cloud synchronization must work transparently without user intervention
- Authentication must provide enterprise-grade security with intuitive user flows
- The application must deploy successfully to distribution platforms with all cloud services operational

Test analysis will result in a comprehensive quality report documenting validation outcomes, performance benchmarks, and any required remediation before final deployment.