# **🛠 Sprint Execution Tasks – Version 1.2 (Daily Devotion & Regional Reach)**

This document outlines the **task-level execution plan** for **version v1.2**, focusing on daily engagement with God’s Word and multilingual support. Each sprint covers frontend, backend, and DevOps activities and is aligned with known risks and performance constraints.

### **✅ Version 1.2 – Daily Devotion & Regional Reach (Oct 11 – Nov 7)**

**Goal:** Build daily engagement with God’s Word and reach multilingual users.

## **🌀 Sprint 6: Oct 11–Oct 24**

**Sprint Goal:** Add Daily Bible Verse + reflection engine

### **✅ Frontend Tasks:**

* Design and implement daily verse UI component (home screen widget)
* Build modal or inline display for short reflection
* Enable navigation from daily verse to a full study guide based on the same verse
* Add push notification prompt + local notification scheduling interface
* Ensure fallback UI for no network or empty verse

### **✅ Backend Tasks:**

* Create Supabase table for daily verses + cached reflections (replacing Firebase collection)
* Implement reflection generation via LLM with static verse schedule
* Enable pre-generation of 30 days of content to reduce live token costs
* Add fallback verse and reflection logic if generation fails
* Define fallback content (e.g., static Bible verse + encouragement message)

### **✅ DevOps Tasks:**

* Create cron job (via Supabase Edge Functions or Firebase Cloud Functions) to generate next-day verse + reflection
* Add timezone-safe logic for daily scheduling (user local time using client-passed offset)
* Set up logging for failed or skipped verse generations

### **✅ Deliverables:**

* Daily verse UI section integrated into home screen
* Pre-generated reflections stored in Supabase
* Notification framework triggered by local device schedule
* Mock mode fallback active during offline or generation failure

### **✅ DoD:**

* Daily verse loads on local time trigger
* Reflection appears with modal or inline experience
* At least 10 days of content tested with LLM and cached
* Telemetry enabled for reflection generation failures

### **⚠️ Dependencies / Risks:**

* Notification system differences on iOS vs Android  
  + **Mitigation**: Use silent push with local scheduling for iOS restrictions
* LLM token consumption for pre-generating large queues
* Clarify fallback logic for when daily verse/reflection is unavailable

## **🌀 Sprint 7: Oct 25–Nov 7**

**Sprint Goal:** Add Hindi/Malayalam support and language switch

### **✅ Frontend Tasks:**

* Add language selector to settings and onboarding screens
* Apply i18n logic to UI strings using flutter\_localizations
* Adjust font rendering for multilingual content (e.g., Malayalam ligatures)
* Ensure TalkBack/VoiceOver screen reader compatibility for localized content

### **✅ Backend Tasks:**

* Modify LLM prompt templates for Hindi and Malayalam generation
* Store language-specific reflections and guides with metadata tags (EN/HI/ML)
* Build fallback mechanism: if multilingual generation fails, fallback to English
* Log language-specific reflection errors with tags (to monitor LLM multilingual issues)
* Clarify if verses are shared across languages or fully localized versions

### **✅ DevOps Tasks:**

* Run prompt tests for Hindi and Malayalam flows with telemetry capture  
  + **Note**: Test at least 10 common verses per language
* Ensure daily cron job supports all active languages
* Configure environment to inject language preference into prompt logic dynamically
* Create alert on repeated multilingual failures via Supabase logs or Slack webhook

### **✅ Deliverables:**

* Hindi/Malayalam reflections generated and displayed based on selected language
* Multilingual prompt templates tested with fallback
* Language switch persisted across sessions (Supabase or local)
* Font rendering verified for Malayalam on multiple OS versions

### **✅ DoD:**

* All study guides respect selected language in both content and UI
* Reflection generator handles Hindi and Malayalam content without crash
* i18n covers minimum 90% of UI strings (non-blocking errors marked for backlog)
* Language-specific failures logged with telemetry or error tags

### **⚠️ Dependencies / Risks:**

* Model hallucination or irreverent translations in regional languages
* Font compatibility across Android/iOS versions for complex scripts
* Clarify distinction between Firebase and Supabase usage for onboarding and i18n persistence  
  + **Note**: Determine whether language preference is stored in Supabase profile or locally