**CAB302 Feature Research for project**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature Research** | | |  |
|  |  |  |  |
| **Feature** | **Technical Requirement** | **Library / Tool / Approach** | **Evidence / Notes** |
| **Login / Save Game / User Progress** | Persistent user accounts, scores, and upgrades | **SQLite** + JDBC or JPA (e.g., Hibernate-lite) | Small embedded database; stores username, password hash, score, and upgrades. Easy to ship with JavaFX apps. |
| **Letter Tile Rack (9 random letters with Scrabble frequency)** | Weighted random generation and UI tile layout | **JavaFX GridPane / HBox** + custom logic | Use a frequency-weighted bag for random letter drawing. Use Button or Label styled as tiles. |
| **Drag & Drop or Click-to-select Tiles** | UI interactivity and custom behavior for tiles | **JavaFX DragEvent** and/or setOnMouseClicked | Use events to allow click/tap to move letters into the word box. Supports both mouse and touch input. |
| **Word Validation (dictionary check)** | Word existence check | **SQLite dictionary table** or **custom word list file** | Load a cleaned word list into an in-memory HashSet or store in SQLite for extensibility. Optionally use Trie for fast prefix validation. |
| **Scoring system using Scrabble values** | Score calculation based on letter values | Simple custom map (e.g., Map<Character, Integer>) | Define static map with letter points. Multiply total by upgrades if applicable. |
| **Submit Word Button** | Score tracking, input handling | JavaFX Button + ActionEvent | Validates word, checks dictionary, updates score, reduces turn count. |
| **Level Progression / Increasing Difficulty** | Level system with escalating target score | Game state model in code | Increase target score, word plays, and difficulty logic stored in model classes (e.g., Level, GameState). |
| **Currency and Shop** | Persistent currency + item unlocking | JavaFX + SQLite or in-memory save | Shop UI using ListView or VBox + simple item models. |
| **Upgrade system (e.g., x2 for double letters)** | Per-run powerups with unique effects | Java inheritance or enum-based upgrades | Upgrades apply post-submission. Example: check if submitted word contains double letters and apply x2 multiplier. |
| **Tooltips / Help UI** | UI explanations for new players | **ControlsFX** or JavaFX Tooltip API | Native JavaFX Tooltip.install(...) on buttons or tiles. ControlsFX enhances visuals. |
| **Word Definitions on Submit** | Lookup word meanings | **Free Dictionary API** or static JSON dataset | Call https://api.dictionaryapi.dev/api/v2/entries/en/<word> or use a bundled definitions file. |
| **Animations / UI feedback** | Visual feedback for player actions | **javafx.animation** or **FXGL** (optional) | Use TranslateTransition, FadeTransition, or FXGL for fancier animations (e.g., tile glow). |
| **Audio (Sound Effects + Music)** | Sound on submit, click, level win | **javafx.media** or **FXGL sound manager** | Play .mp3 or .wav files on certain actions. Use MediaPlayer or AudioClip. |
| **Game Mode Selection / Save + Resume** | Multiple game modes + game state serialization | GameMode enums + JSON or SQLite for save state | Store selected mode + state (letters, score, upgrades, plays left) and resume from it. |
| **Leaderboards** | Shared score ranking | SQLite table or flat file sorted by score | Allow multiple players on the same device. Simple local leaderboard. |
| **Theme Switching (visual or word)** | Skinning or topic-specific word rewards | JavaFX CSS + filtered dictionary | Apply different .css files for themes, or highlight bonus words in selected categories (e.g., “science”). |
| **Daily Challenge / Timed Mode** | Rotating challenge logic | Save last played date + random seed | Store seed per day to generate fixed letter rack. Timer via Timeline. |