**Plugin development plan**

**Objective:**

* Implement a plugin for IntelliJ Ultimate that shows synonyms for identifiers in source code.

**Date: 2023-12-12**

1. **Setup:**
   * Clone the project repository: **git clone https://github.com/GearFifth/intellij-synonyms-plugin.git**
   * Open the project in IntelliJ IDEA Ultimate
   * Ensure that the project builds successfully
2. **Understand Existing Code:**
   * Review existing code in the **toolWindow** package
   * Understand how the tool window works and try it
3. **Cleanup and setup for Java:**
   * Remove the Kotlin directory
   * Remove unnecessary code from **plugin.xml** file (sample code)
   * Create the java directory **src/main/java**
   * Change project structure to use java sdk
4. **Define Intention Action**:
   * Create a simple Intention Action in Java to display a message (SynonymsFinder)
   * Register the Intention Action in the **plugin.xml** file
   * Confirm that the Intention Action is visible and functional
5. **Define SynonymsDialog**:
   * Create a SynonymsDialog that displays synonyms

**Date: 2023-12-13**

1. **Handle thesaurus**:
   * Add **thesaurus.json** to resources
   * Create the ThesaurusRepository for parsing that json.
   * Create the ThesaurusService that injects a repository and calls search function
2. **Implement one-word-synonyms**:
   * Create instance of ThesaurusService inside a SynonymsFinder intention
   * Use it to get synonyms for selected word
   * Display it using SynonymsDialog
3. **Implement multi-word-synonyms**:
   * Define IdentifierSplitter interface
   * Define different splitters that implement IdentifierSplitter interface
   * Create splitterFactory
   * Implement splitter inside ThesaurusService
   * Do the implementation of split method in every splitter

**Problems**:

1. Problem with JAVA language.

Solved by importing library.

1. Problem with description.html

Solved by creating Description.html inside intentionDescriptions.SynonymsFinder

1. Problem with “element instanceof PsiIdentifier” return true for “+, -, =, \*” etc.  
   Solved by creating my own function and checking if it is name of variable, method, class, interface
2. Initially I wanted to use API for thesaurus but I had problem with api subscription

Solved by using local JSON with thesaurus.

1. Problem with too big json file.

Solved by using Jackson Streaming API

1. Problem with multiple word identifiers and different cases.

Solved by implementing splitters for :

* CamelCase 🡪 myVariableName
* PascalCase 🡪 MyClassName
* SnakeCase 🡪 my\_variable\_name
* UpperCase 🡪 MY\_CONSTANT

Every splitter will implement **IdentifierSplitter interface** with split method

I will create **SplitterFactory** that gives me an instance of wanted splitter by case type.

I created **CaseDetector** class with static method detectCase

1. Problem with identifier like this: HTTPResponseCode

Solved by adjusting PascalCaseSplitter to consider consecutive uppercases (acronym)

1. If no synonyms are found use original word for mixing if there is at least one other word with synonyms
2. Possible problem could be that identifier consists of too many words to process.

This could be solved by optimizing method for mixing synonyms and setting some constraints since there could be too many mixed synonyms

**Test examples**

//Single word   
 String floor;

String SERVER;

String window;

//UpperCase

String HTTP\_STATUS\_CODE\_OK;

String API\_KEY\_SECRET;

String SYSTEM\_CONFIG\_PROPERTIES;

//SnakeCase

String database\_connection;

String product\_description;

String payment\_transaction\_log;

//CamelCase

String xmlHttpRequest;

String guiComponent;

String websiteNavigation;

String employeeSalary;

//PascalCase

String ElegantCustomerCar;

String HTTPResponseCode;

String HTMLDocument;