# Proposal for PC Part Picker Application

## Overview

This application is a Java-based desktop program designed to replicate the functionality of the popular website PC Part Picker. It allows users to select and configure components for building a PC, ensuring compatibility between parts, tracking prices, and providing an intuitive and user-friendly interface. The application will be implemented using JavaFX for the graphical user interface (GUI) and MySQL for data management.

## Goals and Objectives

### Core Features:

- Display a catalog of PC components with attributes such as name, category, compatibility, and price.  
- Enable users to select and add components to a build configuration.  
- Validate compatibility between selected components.  
- Calculate the total cost of the configuration.  
- Save and load user configurations from the database.

### User Experience:

- Provide an intuitive and visually appealing interface using JavaFX.  
- Implement responsive features such as filtering and sorting components by category or price.  
- Allow easy navigation between views (e.g., catalog, build summary, and configuration).

### Backend Functionality:

- Use MySQL to manage and store the components database, build configurations, and user data.  
- Provide efficient CRUD (Create, Read, Update, Delete) operations for parts and configurations.

## Implementation Plan

### 1. Database Management

- \*\*Database Design\*\*:  
 - Create a MySQL database named `pcpartpicker`.  
 - Tables:  
 - `parts` (id, name, category, price, compatibility attributes like socket type, power usage, etc.).  
 - `builds` (id, user\_id, part\_ids, total\_price).  
 - `users` (id, username, email, password).  
- \*\*Integration\*\*:  
 - Use JDBC (Java Database Connectivity) to connect the Java application to the MySQL database.  
 - Create utility methods for CRUD operations.

### 2. GUI Design

- Use JavaFX to build the application's user interface.  
- Key Screens:  
 - \*\*Home Screen\*\*: Welcome message, navigation options.  
 - \*\*Catalog View\*\*: TableView displaying PC parts with columns for name, category, and price.  
 - \*\*Build View\*\*: A detailed list of selected components, compatibility warnings, and the total cost.  
 - \*\*Search and Filter Pane\*\*: Sidebar or modal to search for parts by keyword or filter by category and price.  
- Components:  
 - MenuBar for navigation.  
 - TableView for displaying parts.  
 - VBox/HBox for layout organization.  
 - Scene switching for different views.

### 3. Application Logic

- \*\*Component Selection\*\*:  
 - Allow users to add/remove parts to/from the build.  
 - Validate compatibility dynamically (e.g., CPU and motherboard socket matching).  
- \*\*Cost Calculation\*\*:  
 - Automatically calculate and display the total cost of selected components.  
- \*\*Save and Load Configurations\*\*:  
 - Save configurations to the database.  
 - Allow users to reload previous builds.

### 4. Feature Development

- \*\*Search and Filter\*\*:  
 - Provide a text field for keyword search.  
 - Add dropdown menus for filtering by category and sorting by price.  
- \*\*Compatibility Checks\*\*:  
 - Include basic rules for component compatibility (e.g., matching sockets, RAM type, power requirements).  
- \*\*User Accounts\*\* (Optional):  
 - Add user login and registration functionality to save configurations for individual users.

## Required Implementations with Java and JavaFX

### 1. JavaFX for GUI

- `Scene` and `Stage` for managing different views.  
- `TableView` for displaying parts and builds.  
- `MenuBar`, `Button`, `Label`, and other controls for user interaction.  
- `VBox`/`HBox` for organizing layouts.

### 2. JDBC for Database Interaction

- Establish connections to MySQL.  
- Create prepared statements for secure CRUD operations.  
- Handle exceptions for failed queries or connections.

### 3. Java Classes

- \*\*Model Classes\*\*:  
 - `Part`: Represent a single PC part (fields like name, category, price, compatibility attributes).  
 - `Build`: Represent a PC build (list of parts, total cost).  
- \*\*Controller Classes\*\*:  
 - `PartController`: Manage CRUD operations for parts.  
 - `BuildController`: Handle build operations, such as adding/removing parts and saving builds.  
- \*\*Utility Classes\*\*:  
 - `DatabaseManager`: Manage database connections and common queries.

### 4. JavaFX Scene Switching

- Use `BorderPane` or `StackPane` for flexible scene management.  
- Implement methods to switch between the catalog, build, and configuration screens.

### 5. Exception Handling

- Add robust error handling for database operations and UI interactions.  
- Display user-friendly messages for invalid operations (e.g., incompatible parts).

## Project Milestones and Timeline

1. \*\*Week 1\*\*:  
- Set up the MySQL database and connect it to the Java application using JDBC.  
- Create a basic JavaFX UI with navigation between screens.  
  
2. \*\*Week 2\*\*:  
- Implement the catalog view with filtering and sorting.  
- Develop compatibility validation logic.  
- Add functionality to calculate and display the total cost.  
  
3. \*\*Week 3\*\*:  
- Enable saving and loading configurations in the database.  
- Refine the UI for better usability and responsiveness.  
  
4. \*\*Week 4\*\*:  
- Add optional features like user accounts and advanced compatibility checks.  
- Test and debug the application to ensure stability.

## Tools and Technologies

- \*\*Programming Language\*\*: Java  
- \*\*GUI Framework\*\*: JavaFX  
- \*\*Database\*\*: MySQL  
- \*\*IDE\*\*: IntelliJ IDEA or Eclipse  
- \*\*Version Control\*\*: GitHub

## Expected Outcome

The final application will provide users with a robust tool for selecting and configuring PC parts. It will offer features for browsing, filtering, validating compatibility, and calculating costs, with data persistence through MySQL. The intuitive JavaFX-based UI will deliver a user-friendly experience.