# Project Ideas for Software Engineering Course

8. **Atm Simulator System Automated Teller Machine ATM Banking System Java Project**

   The **Automated Teller Machine ATM Banking System** is a banking application developed to perform different banking services through the Automated Teller Machines. The all functions include the regular transactions like cash deposits, cash withdrawals, balance enquiry, balance statements, savings account, and current account; change PIN Number, Credit card Withdrawals and so on. The application design maintains the information of the accounts of various customers including the information of the ATM cards, their types Credit cards, Debit Cards and the transactions done by the customers through the ATM machine centers with co-relation of the Banking Services.

   The stored details also include the information of the various centers in and around the ATM services, which help in the relational maintenance of every transaction in the ATM Machine by the customers with their concerned branch operations.

   The developed application is considered to the version upon the system, which is proposed to be built with the content and touch of the oracle as the centralize database with oracle 9i as the database. The overall banking ATM system is planned to be is the format of distributed architecture as the database platform. The proposals are planed to keep entire architecture to be browser (IE, Mozilla, Chrome) specific.

   The major functions of the overall ATM system are to keep the following component intact.

   - Consistency of the ATM System
   - Integrity of the ATM System
   - Data Security for all customers
   - Data Reliability, Unique and Accuracy
   - User Friendly web pages at admin side and User side
   - To check that the banking ATM system overcome the hurdles of the version specific standards

9. ONLINE CV BUILDER:

   One of the most ambitious java project ideas to consider. This system will take the little information from the user and will provide him with a fully developed CV in return. One can always depend on such a system for preparing CVs at the last minute.

# Personal Finance Side Projects

*These projects will help you achieve a practical goal (get a better handle on your finances), while also improving your software engineering skills.*

10. **A net worth calculator and tracker** (suggested implementation: CLI, web, or mobile app). Build a calculator you can use to track the rise or fall of your net worth on a monthly basis. You can use something like this net worth worksheetfrom Charles Schwab to guide you. Optional extension: have it send you a 12 month report for the previous year on the first of January each year.
11. **A tax forecaster** (suggested implementation: web app). This will be particularly useful if you do any freelance software engineering. Build a tool that takes your freelance earnings as input and then predicts your expected tax liability for the rest of the financial year. Make it smart enough to predict periods of higher or lower demand for your services, and adjust accordingly.
12. **A deal finder** (suggested implementation: web app with mobile notifications). Build a simple web app to notify you when an item you covet goes on sale for a good price. You could use a web scraper to pull the item's product page and notify you of any price changes.
13. **An expense tracker** (suggested implementation: web or mobile app). Create a simple interface you can use to add and categorize your expenses. Generate monthly reports based on the inputs and write custom alerts for things, like, "spending too much money on coffee… as always."
14. **A financial independence calculator** (suggested implementation: web app). Financial independence is, essentially, saving and investing as much of your income as possible so that you don't *need* to work for money. While many financially independent people continue to work, they can now focus on doing work that they love, rather than work that pays the most. Build a tool to calculate, based on your: current savings, investments, income, retirement accounts, and expenses, how far away you are from financial independence. Some examples: FIREcalc, cFIREsim.
15. **A bill splitter** (suggested implementation: mobile app). Build a simple tool to help you and your friends split bills when you go out to eat together.

# Games and Simulation Side Projects

*Most software engineers I know are fascinated by the world of game development, graphics, and simulations, but don't have a lot of experience with them. These projects are small enough that you can set foot into this world without biting off more than you can chew.*

14. **A random name generator** (suggested implementation: CLI, web, or mobile app). Build a random name generator (example) that creates unique names on the fly, based on an algorithm. Use machine learning techniques to help you by training the program with a sample data set of names similar to those you want to generate. Otherwise, create your own lexical rules for how names are generated. For example, a name generating algorithm inspired by The Handmaid's Tale might stipulate that names for Handmaid women should start with 'Of', and end with a random male name, e.g., Ofpeter.
15. **Conway's Game of Life** (suggested implementation: any platform capable of real-time graphical rendering). Conway's Game of Life simulates the lives of simple cells that obey algorithmic laws. This video explains how the game works and includes an example of one possible result:
16. **A procedurally generated map maker** (suggested implementation: browser-based app). Create a browser-based application that allows users to procedurally generate a terrain map based on a random

seed. The map can be as detailed or as simple as you'd like. This project is a good opportunity to learn about procedural generation.

17. **A character generator** (suggested implementation: browser-based or mobile app, CLI). Create a tool that allows you to randomly generate playable characters for your favorite role-playing games, whether they be tabletop games, like Pathfinder, or video games, like Divinity: Original Sin.
18. **Interactive fiction** (suggested implementation: CLI). A fun way to get into game development without having to worry about graphical assets, interactive fiction renders the world for the player through text descriptions. The Dreamhold is a good example of interactive fiction with a useful 'help' command.

# Artificial Intelligence Side Projects

*AI's usefulness in day-to-day software engineering is increasing by leaps and bounds. It's now easier than ever before to make your first forays into the world of Artificial Intelligence.*

19. **An unbeatable Tic-Tac-Toe engine** (suggested implementation: CLI program). For an excellent first AI project, try to write an engine that cannot be beaten at Tic-Tac-Toe. You can achieve this by implementing this strategy, which produces a draw as its worst-case outcome.
20. **A chess engine** (suggested implementation: an engine written in a programming language you want to learn, or master). Try to write an engine that can play chess against a human opponent using a Universal Chess Interfacecompatible GUI, such as XBoard. See Stockfish as an example. For a less daunting challenge, you may wish to focus on the behavior of just one piece, e.g., Knights.
21. **A niche chatbot** (suggested implementation: web app). Some of the greatest minds in the world are working on chatbots that respond in lifelike ways. It's an incredibly difficult challenge, but, by reducing the scope of your chatbot, you have a side project that is more approachable for evenings and weekends. Create a chatbot that produces real-sounding responses based on a niche topic that you're passionate about: your favorite band, video game, sports team, or TV show. Leverage an existing library to help you, such as ChatterBot.
22. **A spam classifier** (suggested implementation: any programming language you want to master). Build a tool to classify whether an email is or isn't spam based on the content alone. You can use this public data set of emails from the Enron investigation to test your spam classifier.

# Entertainment Side Projects

*These projects are for software engineers who want to work on something fun and light-hearted.*

23. **A movie showtime finder** (suggested implementation: web or mobile app with email or text message notifications). Build a program that notifies you, by text or email, about showtimes for potentially interesting movies playing at your favorite cinema. The concept of an 'interesting movie' can be derived using machine learning (if you watch enough movies to have good training data), or a handcrafted algorithm. For example, you might use the Open Movie Database API, paired with an HTML parser like BeautifulSoup, to build a program that alerts you to sci-fi movies rated 7.0 or above on IMDB, movies starring Amy Adams, and/or any movie with an average rating of 8.0 or higher.

24. **A spoiler blocker** (suggested implementation: browser extension). Short of implementing a total social media/internet/watercooler chat ban, it can be difficult to avoid seeing spoilers for your favorite TV show… especially if that TV show is as popular as Game of Thrones. Create a browser extension that removes all mentions of your favorite show from any webpage loaded in your browser… or replaces them with cute pictures of puppies.

# Fun Side Projects

*These projects are all different, from logging sensor data to finding new desktop backgrounds for your computer. One thing they all have in common is that they're fun, and several of them will have you work with interesting APIs.*

25. **Pixel art generator** (suggested implementation: any programming language you want to master). Build a tool that takes an image as input and samples the image to produce pixel art as output. If you want to improve your front-end skills, generate the resulting pixel art using CSS.
26. **Music suggestion tool** (suggested implementation: build a wrapper for the Spotify API). Create a tool that tracks the music you listen to and generates a playlist with similar qualities, but of songs you haven't heard before. The Spotify API provides all of the tools needed to extract what you've listened to and create a playlist — the recommendation engine is up to you!
27. **Temperature logger** (suggested implementation: Raspberry Pi, temperature sensor, web app). Have your Raspberry Pi hooked up to a temperature sensor and send temperature data from your home to a web app that saves the data to a database or updates a CSV file. For bonus points, have a weekly temperature report emailed to you. Other Raspberry Pi projects: home automation, home security.
28. **Microlearning app** (suggested implementation: web or mobile app). Build an application that sends you one page per day about something you want to learn. This could be a random page from Wikipedia, a page of Reactdocumentation, a Kanji character, or a page from the CIA World Factbook.
29. **Slack bot** (suggested implementation: Slack API). If you or your team use the popular chat app Slack, build a bot to make some aspect of your (or your team's) life easier. Ideas: a coffee order bot, a bot that reports daily on the number of commits made to your team repos, or a daily stand-up reminder bot.
30. **Daily desktop background** (suggested implementation: Unsplash API, scripting language for your OS). Build an app that refreshes your desktop background with a new image every day.